Escuela Politécnica Superior de Jaén
Máster Universitario en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior (Jaén)

Trabajo Fin de Máster

DETECCIÓN DE EMOCIONES A PARTIR DE LA ENTONACIÓN DEL INTERLOCUTOR

Alumno/a: **Hortelano Ruiz, Francisco Javier**

Tutor: Prof. D. Manuel Carlos Díaz Galiano
Tutora: Prof. Dña. Pilar López Úbeda
Dpto.: Departamento de Informática

Julio, 2020

Universidad de Jaén
Escuela Politécnica Superior de Jaén
Departamento de Informática

Don Manuel Carlos Díaz Galiano y Doña Pilar López Úbeda, tutores del Proyecto Fin de Máster titulado: Detección de emociones a partir de la entonación del interlocutor, que presenta Francisco Javier Hortelano Ruiz, autorizan su presentación para defensa y evaluación en la Escuela Politécnica Superior de Jaén.

Jaén, 6 de JULIO de 2020

El alumno:

Francisco Javier Hortelano Ruiz

Los tutores:

Firmado por DIAZ GALIANO MANUEL CARLOS - 26032999N el día 06/07/2020 con un certificado emitido por AC FNMT Usuarios

Manuel Carlos Díaz Galiano

Firmado por LOPEZ UBEDA PILAR - 15451728Y el día 06/07/2020 con un certificado emitido por AC FNMT Usuarios

Pilar López Úbeda

AGRADECIMIENTOS

En primer lugar agradecer a mi familia el apoyo recibido todo este tiempo, por haberme enseñado a dar siempre un poco más, a no rendirme y a saber cambiar de rumbo si algo no funciona. Por haberme educado y enseñarme que la constancia y el esfuerzo son los mejores ingredientes para conseguir lo que te propones. He conseguido llevar a cabo este proyecto gracias a vosotros.

A Marta, por haber seguido dándome fuerzas y ánimo sin importar las condiciones. Gracias por ser una pieza esencial en mi vida, por tu generosidad y por ayudarme a ver las cosas más claras cuando parecían nubladas.

A mis tutores, Manuel Carlos y Pilar: por aceptar este proyecto que les propuse, por la confianza depositada en mi para hacerlo, por darme libertad para experimentar y descubrir el desarrollo del proyecto; por dirigirme y reconducirme si me desviaba y por el empeño puesto al orientarme y corregirme. He aprendido mucho de vosotros, profesional y personalmente, y os considero un ejemplo a seguir.

A toda la gente que ha colaborado con este proyecto desinteresadamente, ya sea a la hora de grabar audios, de recomendarme artículos o libros relacionados con el proyecto, o a ofrecerme ideas y enfoques. Habeis sido la base para el proyecto y, sin vuestra ayuda, no lo habría conseguido.

En último lugar, agradecer a todos los profesores del Máster de Ingeniería Informática que han contribuido en esta etapa de educación y formación profesional en la Universidad de Jaén.

A todos, muchas gracias.

Índice

Capítulo 1

Introducción

1.1. Estado del arte / Antecedentes

Una de las principales ramas que se desarrolló conforme se sofisticaban y potenciaban los ordenadores, fue la interacción persona-ordenador. Comenzando por simples tarjetas perforadas, y pasando por simples interfaces, como la consola de comandos, fueron surgiendo estudios sobre la relación persona-ordenador. El primero de estos, fue escrito por Joseph Carl Robnett Licklider, en 1960 [1]. Este autor, junto con Welden E. Clark [2] elaboraron una lista de diez necesidades que deberían ser cubiertas a corto, medio y largo plazo. Entre las medidas que debían tomarse a largo plazo, las dos primeras se refieren a la comprensión del lenguaje natural escrito y hablado, áreas que estudiaría más tarde la rama de PLN (Procesamiento del Lenguaje Natural).

1.1.1. Emociones y PLN

Para hablar de las emociones en el ámbito de la informática, es necesario hacer mención del PLN . El PLN [3] es un campo de investigación y aplicación que explora cómo los ordenadores pueden ser usados para entender y manipular el lenguaje natural escrito y hablado para realizar acciones útiles. Aunque aún no está completamente desarrollada, es un campo muy joven que apenas sobrepasa los veinte años de vida, su integración en las interfaces de usuario, entre otras aplicaciones, en los últimos años ha sido más que notable.

En la temática que compende al proyecto, se centra en el PLN del diálogo. Principalmente está centrada en traducir al "lenguaje máquina" el texto que se extrae del habla del usuario, para manejar la aplicación en cuestión y realizar las acciones pertinentes. Pero al centrarse únicamente en las palabras pronunciadas, se deja de lado otra información importante que se transmite, que es la entonación. Una de las líneas de investigación que toca el PLN es el tratamiento de las emociones, muy utilizada sobre todo a la hora de valorar opiniones de los clientes sobre un producto.

No obstante, en la detección de emociones, una gran parte de la información de la comunicación verbal reside en la entonación del habla. La entonación de una frase puede cambiar totalmente el significado de la misma (como es el caso de la ironía o el sarcasmo), indicar un estado de ánimo adicional al de la frase (como nerviosismo), etc. Por consiguiente, tanto en una interacción con un ordenador como en la ampliación del ámbito de aplicación de la detección de emociones, debe tenerse en cuenta la entonación. Un ejemplo práctico, sería la adecuación de mensajes mostrados al usuario en función de si se detecta enfado en el habla, para mitigar ese enfado.

Uno de los retos a los que se enfrenta el PLN es el área de las emociones; tanto para su detección como para su tratamiento en distintos campos, como la negación, análisis de sentimientos, etc. Según Ali Yadollahi, Ameneh Gholipour Shahraki y Osmar R. Zaiane [4] el análisis de sentimientos se divide en la Minería de Opiniones y en la Minería de Emociones. Esta última categoría comprende varios campos que son:

- Detección de emociones
- Clasificación de la polaridad de la emoción
- Clasificación de emociones
- Detección de la causa de la emoción

En el caso que ocupa al proyecto, se clasificaría dentro de la detección de emociones principalmente, aunque puede tratar aspectos de la clasificación de emociones. Por tanto, se explicará en qué consiste cada disciplina para tener una mejor compresión del tema que se aborda. La detección de emociones consiste en detectar si un texto transmite algún tipo de emoción o no. Esta tarea es muy similar a la detección de subjetividad en opiniones. En cambio, la clasificación de las emociones consiste en la clasificación detallada de la emoción presente en un texto, dentro de un conjunto definido de emociones.

Esta agrupación de las emociones se ha estudiado desde los años 60. Prevalecen dos teorías: la teoría de emociones discretas y el modelo dimensional. La primera afirma que las diferentes emocionnes provienen de sistemas neurológicos separados, mientras que la segunda propuesta sostiene que un sistema neurofisiológico común e interconectado es respondable de todos los estados afectivos. De esto surge un nuevo concepto; las emociones básicas. Estas emociones básicas se refieren a aquellas que no se componen de otras emociones y que pueden ser reconocidas por las personas alrededor del mundo independientemente de su raza, cultura y lengua. Así pues, se pueden distinguir en distintos modelos propuestos por los teóricos de este campo:

- Ekman (1972). Su modelo es discreto y contiene las emociones básicas: enfado, asco, miedo, alegría, tristeza y sorpresa [5]. Varios años después lo amplió a 12 emociones básicas positivas y negativas [6], pero se suele seguir usando su clasificación inicial.

- Plutchik & Kellerman (1986). La clasificación que estos autores sostienen, agrupa emociones de cuatro ejes bipolares, "enfrentando" las emociones, por oposición: alegría y tristeza, miedo y enfado, confianza y asco, y sorpresa y anticipación. Estas emociones son triviales en algunos casos, como alegría y tristeza, pero en otros cuesta más hacer una distinción, como es sorpresa y anticipación [7].

- Shaver (1987). Esta agrupación es construida en una estructura de árbol en la que se sostiene el principio de que cada emoción básica es una de las ramas principales del árbol y cada una tiene su propia categorización [8].

- Lövheim (2011). Sugiere un modelo dimensional que se basa en una vectorización de cada emoción en función de tres hormonas: serotonina, noradenalina y dopamina. Cada emoción es un vector de tres dimensiones, con valores de 0 a 1, en función de la cantidad de hormonas segregadas para cada una. Así, se forma una estructura de cubo en el que las emociones básicas representan los vértices de dicho cubo [9].

Para este proyecto, se ha escogido la clasificación inicial de Ekman, añadiendo el estado neutro, obteniendo así un modelo con 7 emociones: alegría, asco, enfado, miedo, sorpresa, tristeza y neutro.

1.1.2. Procesamiento de audio

El procesamiento del sonido surge cuando, conociendo las propiedades físicas de las señales, pasa a transmitirse digitalmente y es necesario aplicarle un procesamiento para codificar y decodificar la señal, entre otros aspectos . De hecho, la acústica es la disciplina física encargada del estudio del sonido. Pero más concretamente, en el caso que nos ocupa, se centra en el procesamiento del audio orientado al habla, conocido como Speech Processing.

Esta rama consiste en el estudio de las señales del habla y los métodos para procesar dichas señales. Normalmente tratadas mediante su representación digital, el procesamiento del habla puede considerarse un caso especial del procesamiento de señales de audio aplicado a las señales del habla y la voz. Este campo, a su vez, puede dividirse en varias disciplinas:

- Reconocimiento de voz, para extraer la información lingüística de una señal de voz.
- Reconocimiento de locutores, para identificar al hablante.
- Mejora de los valores de la señal, aumentando la calidad mediante técnicas de limpieza de la señal. Un ejemplo de esto es la reducción de ruido.
- Análisis y síntesis de voz.

Estas técnicas trabajan generalmente sobre los atributos que definen al sonido producido por el habla. Este sonido, al igual que el resto de ellos, se caracteriza por los atributos conocidos como cualidades del sonido [10] . Estas cualidades son:

- **Frecuencia.** Conocida también como la altura del sonido, consiste en la afinación del sonido. Físicamente hablando, es el número de oscilaciones de una onda en un período de tiempo (normalmente 1 segundo), y se mide en hercios (Hz). En base a esta cualidad, se pueden distinguir sonidos graves y agudos. Los sonidos graves tendrán una menor frecuencia y los agudos mayor frecuencia. El rango de frecuencias que el ser humano puede escuchar está entre 20 y 20.000 Hz.

- **Intensidad.** Relativo al volumen del sonido. Directamente proporcional a la amplitud de la onda, permite distinguir entre sonidos débiles (con una menor amplitud de onda) y fuertes (con una mayor amplitud de onda). Esta cualidad se mide en decibelios (dB) y el rango de estos decibelios va desde 0, donde el sonido no sería audible por el oído humano, hasta los 140, donde se considera el umbral del dolor auditivo.

- **Duración.** Esta cualidad indica el tiempo que una onda de sonido mantiene su vibración. Depende de la longitud de onda y diferencia entre sonidos largos y breves. En el habla, la duración del sonido depende del tiempo que la persona esté hablando.

- **Timbre.** Es la cualidad del sonido que diferencia a la fuente del sonido. Con las 3 primeras cualidades se obtendrían ondas invariantes a factores ambientales. Una onda así no vería modificadas ninguna de las cualidades anteriores, independientemente de cualquier factor externo. Pero el sonido vibra de diferente manera dependiendo de muchos factores: el material que lo produce (madera, metal...), el medio por el que se propaga (aire, líquido, sólidos con distinta densidad...), etc. Así pues, dos instrumentos distintos, como una flauta dulce y un trombón, al dar la misma nota musical, suenan de manera diferente. De igual manera, este fenómeno es el que provoca que cada persona tenga una voz distinta, obteniendo voces dulces, roncas, rasgadas, etc. El timbre es modificado según se modifiquen las 3 cualidades anteriores. Generalmente, lo que hace a cada fuente de sonido característica y en nuestro caso la voz humana, es una serie de ondas

características que vibran simultáneamente. Es decir, los sonidos complejos como la voz, están compuestos de varias ondas sonoras (cada una con una frecuencia e intensidad concretas) que dependiendo de estos dos factores y del número de armónicos de cada onda, forman el sonido característico reconocido como timbre.

Cada temática mencionada del procesamiento de la voz, se fija en unas cualidades del sonido u otras, dependiendo de su función. Así pues, el reconocimiento de locutores, extraerá las características del audio que mejor puedan definir el timbre del sonido, facilitando así la identificación de la persona en cuestión.

Para el presente proyecto se utilizarán diversas técnicas, pertenecientes a algunas de las disciplinas mostradas anteriormente para el procesamiento digital del audio, dependiendo de la cantidad de información, de cada una de las cualidades del sonido, que proporcionen. Asimismo, no solo se tendrá en cuenta cuánta información, de cada cualidad, proporcionan, sino también cómo se ofrece, qué características tiene y cómo de representativa es.

1.1.3. Emociones y Audio

En cuanto a las emociones producidas en la voz, hay que tener en cuenta que se pueda realizar una diferenciación (parámetros y características sonoras) de cada emoción. Esto queda demostrado por Felix Burkhardt y Walter F. Sendlmeier [11] donde muestran, tras una serie de experimentos, que las anteriores cualidades del sonido varían para cada emoción descrita. En su estudio utilizan el conjunto de emociones formado por enfado, alegría, felicidad, tristeza y aburrimiento. Además comprueban que hay emociones que suelen confundirse más frecuentemente que otras, como el caso de alegría y felicidad, o una tristeza más sosegada con el aburrimiento. Un punto interesante que tratan en el artículo es que realizan una diferenciación en el enfado y la tristeza. El enfado lo dividen en hot anger y cold anger, lo que se podría traducir como cólera y enfado contenido; y la tristeza la dividen en crying despair y quiet sorrow, traducido como llanto desatado y tristeza sosegada. Esta diferenciación es realizada porque la emoción puede variar según los estímulos que la provoquen, resultando en

una gama de matices dentro de la misma emoción, esto es debido a la **expresividad**.

La expresividad matiza una emoción u otra haciendo que se intensifique dicha emoción o se atenúe. Este concepto es tratado por William Roberts y Janet Strayer en su artículo Empathy, Emotional expressiveness and Prosocial Behaviour [12], en el que sostienen que existen varios factores a la hora de expresar una emoción que afectan a la expresividad con la que se manifiesta. Con expresividad se refiere, en este caso, al énfasis con la que es producida dicha emoción. Así pues, una emoción se puede decir de manera que, o bien la emoción queda bien clara, o bien más neutra, dependiendo de la situación en la que se diga. Este es el caso del estudio anterior, en el que diferencian entre dos tipos de enfado o de tristeza.

Como se ha mencionado anteriormente, hay situaciones en las que, por el tono en el que es dicha una frase, puede confundirse con una emoción concreta, sin estar expresándola el interlocutor. Por ejemplo, una orden dicha con tono firme puede confundirse con un tono de enfado sin ser la intención del hablante. Esto es debido a que las cualidades del sonido de dichas emociones, cuando son exageradas (hacia arriba o hacia abajo) son muy similares a los atributos de otra emoción. Además, todas las emociones pueden tender hacia la neutralidad, cuanto menor sea la expresividad con la que se dice. Para casos similares a estos o en los que, simplemente se dice una frase sin emoción, se introduce el estado neutro en el conjunto de emociones que se tratarán en este proyecto.

1.1.4. Trabajos existentes

Existe una gran cantidad de trabajos relacionados con este área, probando una infinitud de técnicas, centradas sobre todo en el ámbito del Machine Learning (ML) y el Deep Learning (DL). Wahab Khan et al. (2016) [13], muestran en su estudio la gran cantidad de técnicas de ML utilizadas en PLN, en el que indican, además, las ventajas de utilizarlo. Un ejemplo de esto es el uso de clasificadores o SVM (Support Vector Machines) para la clasificación binaria de datos, como es la polaridad de una frase. No obstante, para realizar datos y extracción

de información más compleja de los mismos, se están empezando a aplicar técnicas de DL también, gracias al desarrollo de las redes neuronales, siendo las LSTM (que trataremos más adelante en la memoria) las que mejor se adaptan para los problemas de PLN. Algunos trabajos [14], han sido una gran referencia en marcar caminos para seguir. En estos 15 años, debido al auge, se han creado softwares comerciales que realizan esta representación de emociones. El software creado por Petrushin [15] en 2007 fue de los primeros en salir al mercado realizando esta tarea con gran acierto y cobrando relevancia. Unifica los procesos seguidos por los estudios que cimentaron las bases de este campo, y es tomado como referencia por muchos estudios posteriores por el proceso seguido y el método de extracción de características.

Cabe destacar el trabajo realizado por los investigadores españoles Iker Luengo, Eva Navas, Inmaculada Hernáez y Jon Sánchez [16] por su artículo Reconocimiento automático de emociones utilizando parámetros prosódicos, publicado dos años antes del lanzamiento de la aplicación mencionada anteriormente. En su estudio utilizan SVM y GMM para clasificar las emociones del euskera, tomando el mismo conjunto de emociones que se escogen para este proyecto, y con unos resultados que igualan, e incluso mejoran en algunos casos, a los del software anterior.

Este es un campo que está creciendo cada vez más y que muchos investigadores empiezan a combinar con otras áreas para conseguir resultados más completos. Es por esto que hay estudios que combinan lo anterior con el procesamiento del texto, consiguiendo así un software que combina la emoción detectada en la información lingüística y en la información sonora para determinar la emoción final expresada. Gracias a enfoques como este, puede aumentar el campo de aplicación y estudio considerablemente.

No obstante, en este proyecto se centrará únicamente en la clasificación a partir de la información sonora, sin tener en cuenta la el contenido emocional lingüístico.

1.2. Propósito general y objetivo

El objetivo de este proyecto consiste en la creación de un software que permita obtener la emoción básica expresada por una persona, teniendo en cuenta únicamente la entonación con la que habla. El idioma para el que se construye el sistema es el castellano, clasificando así en base a cómo se expresen las emociones en este idioma. Las emociones en las que podrá clasificar serán del conjunto definido por las 7 emociones elegidas:

1. Alegría
2. Enfado
3. Asco
4. Tristeza
5. Miedo
6. Sorpresa
7. Neutro

Además deberá emplearse el uso de redes neuronales, permitiendo la experimentación algunos tipos de todos los existentes , para comprobar cuál consigue mejores resultados o cuál se adapta mejor al problema. Las redes neuronales se escogerán en base a las características que se extraigan, según la información que extraigan del audio, su estructura, organización y disposición de los datos.

El sistema, a su vez, deberá cumplir los siguientes requisitos:

1. **Robusto a ruido.** Deberá funcionar incluso aunque la claridad del audio no sea bueno, o esté en un ambiente en el que hay más ruido.
2. **Robusto a las variaciones de las cualidades del sonido.** No dependerá de si la persona es hombre o mujer (timbre y frecuencia), de mayor o menor edad, hable más o menos rápido; el sistema debe poder abstraer la emoción.

3. **Robusto a variaciones de expresividad.** El sistema debe poder agrupar en la misma emoción los audios que la muestren con una expresividad mayor (más enfatizada) y con menor expresividad (emociones más neutras).

1.3. Metodología

La metodología seguida para este proyecto ha consistido en un enfoque retroalimentado. Se ha comenzado por un estudio inicial del estado del arte para conocerlo y ver las distintas maneras de afrontar dicho problema. A partir de aquí, y tras la construcción de la base de datos con los audios que servirían para los experimentos, se ha seguido el siguiente proceso:

1. **Hipótesis:** Se plantea la hipótesis de cómo podría funcionar el sistema. Se define el marco teórico de dicha hipótesis para desarrollarlo en los siguientes pasos.

2. **Características extraídas:** En base a la hipótesis, se estudia qué características hay que extraer de los audios y cómo se pueden extraer.

3. **Procedimiento aplicado:** Se define el tipo de red neuronal con su configuración y el tipo de entrenamiento definido para la misma.

4. **Experimentos:** Se realizan varios experimentos, con pequeñas variaciones entre ellos, como el nº de características, añadiendo o quitando algunas, cambiando el algoritmo de entrenamiento, modificando la configuración de la red neuronal, etc.

5. **Resultados y conclusiones:** Tras los experimentos, se analizan los resultados obtenidos y se extraen conclusiones: qué ha funcionado, qué no ha funcionado y por qué.

6. **Redirección:** Tras responder a las 3 preguntas anteriores, se tiene en cuenta lo que no ha funcionado y el por qué, para desechar posteriores nuevas hipótesis que contengan dicho error.

Este proceso se ha repetido hasta obtener unos resultados satisfactorios, y por tanto, en esta memoria se verá reflejado el mismo orden a la hora de presentar el trabajo realizado con

los experimentos.

Los capítulos, que a continuación de este comienzan, muestran el proceso seguido durante todo el proyecto, así como los pasos detallados para poder ser reproducido por el lector de este trabajo. Se comienza definiendo el corpus utilizado en el proyecto, desde su estudio y planteamiento hasta su construcción, y las condiciones del entorno utilizadas en todos los experimentos. Tras esto, se define, en cada uno de los siguientes capítulos, un experimento en el que se incluye la estructura definida en los párrafos anteriores. Cada experimento partirá del resultado y conclusiones del experimento, y por ende capítulo, anterior. Finalmente, se realizarán unas conclusiones generales del proyecto, así como los trabajos futuros a los que deja paso el trabajo, y una valoración personal del proyecto.

Capítulo 2

Corpus de audios

Una vez introducido el proyecto, en el capítulo se detallará todo el proceso seguido para generar el corpus utilizado en el proyecto, partiendo de un estudio de los corpus existentes, características que debe tener el necesitado para este estudio, y la formación de dicho corpus, pudiendo ser reproducible siguiendo los pasos descritos.

2.1. Antecedentes

Para poder construir el sistema, primero es necesario generar una base de datos, o corpus, de la que partir. Este corpus estará constituido por audios grabados a personas expresando las distintas emociones y será usado para entrenar y validar el sistema. Además debe cumplir varios requisitos:

1. **Mismo número de audios para cada emoción.** Una condición básica pero necesaria para evitar que el sistema tenga tendencia hacia una emoción, al haber recibido un sobre-entrenamiento en dicha emoción por existir más muestras. Tal y como muestran en sus experimentos Batyrshin y González Mendoza (2012), un corpus balanceado, suele proporcionar mejores resultados que otro que no lo esté. Por tanto, la cantidad de audios por emoción, ha de ser idéntica en cada caso. [17]

2. **Mismo número de audios de cada persona para cada emoción.** Si la base de datos ha sido creada por audios de distintas personas, debe haber el mismo número de

audios por emoción grabados por cada persona. Esto garantiza que las variaciones de las voces sean iguales para cada emoción. En otras palabras, si se graba a cada persona para algunas emociones en concreto, en vez de para todas, las variaciones en la entonación podrían quedar ocultas por las características de las voces. Esto podría significar que dependiendo de las características que se extraigan de la base de datos, pudiera afectar a la clasificación y se confundan emociones debido a que las voces sean más agudas o graves, pudiendo derivar en una identificación del locutor, más que en un reconocimiento de emociones. También podría afectar a la validación en una emoción de la que solo se tienen muestras con voces graves: posteriormente al clasificar un audio con la misma emoción pero con voz aguda no sería reconocido.

La mayoría de artículos que tratan esta temática no hablan de la base de datos utilizada. La información que ofrecen sobre la misma es que está compuesta por una batería de audios grabados a gente que reflejan esas emociones. En estos casos algunos especifican que han hecho uso de un locutor y/o locutora (dependiendo de si la base de datos contempla audios de ambos géneros o no) para grabar los audios. Pero no hacen alusión a cómo está formado el corpus de frases que usan para grabar a los locutores.

La respuesta más útil aparece en el artículo publicado por Planet García, Morán Moreno y Formiga Fanals en 2006 [18], que crean su propia base de datos. En su artículo muestran que el corpus utilizado tiene las siguientes características:

- Contiene 200 frases en castellano grabadas por un locutor no profesional.
- Estas 200 frases están repartidas entre cuatro estados de ánimo que usan en su clasificación: alegría, tristeza, enfado y estado neutro. Cada emoción tiene 50 frases dedicadas.
- Las frases escogidas tienen un contenido semántico que no induce ninguna emoción en concreto.

Sin embargo, hay algunos artículos que hacen uso de bases de datos que ya existentes. Algunas de estas bases de datos y corpus son:

- **Berlin Database of Emotional Speech.** Base de datos de 500 audios en alemán que reflejan las emociones de felicidad, enfado, ansioso, temeridad, aburrimiento y decepción [19].

- **RekEmozio.** Base de datos de material audiovisual en castellano y vasco grabado por 10 actores para el castellano y 7 para el vasco, tomando la clasificación de Ekman [20].

- **CREMA-D: Crowd-Sourced Emotional Multimodal Actors Dataset.** Conjunto de 7.442 clips de audio de 91 actores sobre 10 frases, en distintos estados de ánimo: enfado, asco, miedo, felicidad, tristeza y neutro [21].

- **RAVDESS: The Ryerson Audio-Visual Database of Emotional Speech and Song.** Base de datos de audio y canciones, grabados por 24 actores profesionales equilibrados en cuanto al género, y que contempla las emociones de Ekman y además el estado calma [22].

- **MSP-Improv.** Base de datos audiovisual grabada por 12 actores, en las que se toman muestras de las reacciones de estas personas a distintos estímulos [23].

No obstante, de estas bases de datos, la única accesible gratuitamente es la base de datos CREMA-D grabada en inglés, la cual no es válida para este proyecto, puesto que el idioma objetivo para el que se construye el sistema es el castellano. Por tanto, la decisión tomada para comenzar el proyecto es la creación de una base de datos propia.

2.2. Creación del corpus

A la hora de diseñar la base de datos, se va a seguir un planteamiento similar a los descritos por los estudios que forman la suya propia también. Se plantean varias frases que serán grabadas posteriormente por personas reflejando las 7 emociones propuestas para este proyecto. Para ello se atenderán a los siguientes pasos, que aquí se definen, y se desarrollarán en las secciones siguientes:

1. Selección de las frases, con las características de su contenido lingüístico, utilizadas para generar los audios.

2. Selección de la población para generar la grabación de audios.

3. Condiciones para la grabación de los audios.

4. Estructuración y organización de los audios en el corpus.

2.2.1. Frases

Las frases que se usarían se organizaron en conjuntos de 20 frases para cada emoción. La mitad de estas (10 frases) tendrían un contenido semántico acorde a la emoción asociada, mientras que las restantes tendrían neutro dicho contenido. De esta manera, para la emoción de asco, por ejemplo, habrá 10 frases que expresen asco y otras 10 que no expresen emoción alguna. Esto hace un total de 140 frases que grabar.

Además, las frases neutras serán iguales para todas las emociones, dicho de otra forma, 10 de las 20 frases de cada emoción son comunes a todas las emociones. Esta decisión está basada en que una frase, con un cierto contenido semántico, propicia una expresión más natural de la emoción que refleja. Esto permite que la persona pueda expresar una emoción con mayor énfasis y más fácilmente, lo que consigue una reacción más natural. Por otro lado, al tener las frases neutras, se consigue tener una expresión de la emoción más moderada. Así se cumple con un rango de expresividad más grande dentro de cada emoción.

Las frases están tomadas de sitauciones cotidianas, fragmentos de películas, libros, etc. que permitan a las personas conectar mejor con la emoción y conseguir una reacción espontánea y natural. Estas frases han sido validadas por las mismas personas empleadas en la grabación, matizando sobre todo aquellas que no conseguían un contenido semántico neutro, de manera que no evocara ninguna emoción por mínima que fuera. Estas frases pueden verse en el Anexo I.

2.2.2. Grabación

Para la grabación de dichas frases no se ha tomado a locutores profesionales, sino a personas sin ningún tipo de cualificación en la actuación o locución , de manera que se consiguiera más el factor de naturalidad. Se tomaron las voces de 12 personas, en las que había 6 mujeres y 6 hombres. Las edades de estas personas varían de los 18 a los 60 años, y además de distintas zonas demográficas de España. A priori, lo que se pretende conseguir con esta variedad es cubrir las distintas formas de expresar una misma emoción, lo que debería concluir en un sistema más robusto. Estas variaciones se pueden ver reflejadas en casos como los siguientes:

- La velocidad con la que se habla, dependiendo principalmente de la edad. La gente joven tiende a hablar más fluida y deprisa que la gente con más edad.
- El volumen con el que se habla, dependiente de la edad, el acento, la zona geográfica en la que viva, etc.
- Voces graves y agudas, tanto de hombres como de mujeres. Aunque en este proyecto no se va a realizar una diferenciación entre géneros,

A la hora de grabar las frases debían expresar, de la manera más natural posible, la emoción correspondiente a cada frase. Para ello se recurre a una inducción emocional en la persona [24], previa a la grabación de dicha emoción. Esto consiste en ponerle situaciones, objetos o eventos (estímulos) a cada persona que le evoquen un estado de ánimo que propicie a la expresión de dicha emoción. Debido a que no todas las personas grabadas residían en la misma ciudad, esta ambientación corría a cuenta de cada una, de manera que se hiciera de forma más precisa.

Al ser esta parte del proceso más personal y diferente a cada persona, no se ha hecho una guía de todos los métodos utilizados específicamente detallados, pero entre los más frecuentes se encuentran:

- **Aprovechar una situación no provocada.** Aunque haya estudios que lo avalen, es complicado concienciarse para conseguir una expresión emotiva genuina; siempre quedan trazas que denotan que es forzada. Por tanto la mayoría ha optado por aprovechar cuando

estaban en dicho estado de ánimo (conseguido por un estímulo natural) para grabar las frases correspondientes a dicha emoción.

- **Acudir a estímulos usuales.** A la hora de provocarse una emoción concreta, ocurre que, dicha emoción, no siempre es sencilla de conseguir en el día a día, como por ejemplo la sorpresa, el asco o el miedo. Para conseguir una emoción lo más real posible, recurren a objetos, eventos, etc. que ya conozcan previamente que les produce dicha emoción. Así, aunque provocada, se consigue una emoción más fiel a la real o incluso verdadera.

Para garantizar que no se producía un solapamiento de emociones por grabar todas del tirón y cambiando de emoción drásticamente de una a otra, se pidió la grabación de cada emoción en distintos días. Ya que las personas no son actores profesionales y se pretende conseguir una emoción lo más natural posible, cambiar en un mismo día y de manera inducida el estado de ánimo podía no proporcionar los resultados deseados. Por ese motivo, se indicó que si un día se grababan todas las frases de una misma emoción, entonces no se volviera a realizar la operación hasta mínimo el día siguiente.

La grabación al no disponer todo el mundo de grandes medios técnicos, se realizó con la grabadora del dispositivo móvil de cada persona. Las grabadoras suelen grabar en formato .wav, .ogg o .mp4, de manera que se conseguía un sonido con buena calidad. Además, se pidió que para grabarlas hubiera el menor ruido ambiente posible.

Esta grabación podían enviarla en un mismo archivo de audio con todas las emociones en él o un archivo de audio por cada frase y emoción. En el primer caso, para evitar que fueran archivos muy pesados o se dijeran las frases de manera atropellada una detrás de otra, se les indicó que al grabar una frase, pusieran la grabación en pausa antes de continuar. De esta forma, podían mentalizarse de la siguiente frase, ponerse en contexto, ensayarla para no equivocarse, etc.

2.2.3. Estructura del corpus

Una vez recopilados los audios, la organización de los mismos se procede de la siguiente manera:

1. Mediante el programa de edición de audio **Audacity** se editan los audios recibidos de diferente manera, según si contiene un mismo archivo todas las frases o si es un archivo por frase. Si el archivo tiene una sola frase, se recorta el audio por el principio y por el final, de manera que haya aproximadamente 100 milisegundos de silencio (o ruido ambiente, dependiendo de la calidad de la grabación) por el principio y por el final. Si el archivo contiene varias frases, se separa y procesa cada frase por separado, guardándola en un archivo que contenga las condiciones anteriores, incluyendo los 100 ms de silencio iniciales y finales.

2. Una vez es recortado el archivo, se decide pasar toda la información a mono. Existen audios en mono y en estéreo, y esto debe ser unificado para poder ser procesado de igual manera. La diferencia principal entre mono y estéreo es el número de canales que componen el audio. Normalmente el estéreo se utiliza para codificar la información en dos canales (derecho e izquierdo) para que luego sea reproducido el sonido de cada uno por el altavoz correspondiente. En el caso de existir un único altavoz la información se codifica en mono y se reproduce por él.

   No tiene mucho sentido tener un audio de entrenamiento en estéreo por varias razones:

   - El canal es irrelevante. En el caso de estudio que nos ocupa, no se analiza si la información es mayor en el canal izquierdo que en el derecho o viceversa. La información viene codificada en ambos canales.
   - La codificación de estéreo a mono se produce sumando las frecuencias en cada muestra en cada canal y realizando la media, es decir, al codificar a mono se consigue reducir la dimensionalidad de los datos, sin perder información y eliminando la información irrelevante como el canal que más energía tiene en la onda.

- Los audios que estuvieran en mono, habría que convertirlos en estéreo. Realizando a la inversa el proceso anterior, resultaría en dos ondas iguales a la original, una en cada canal, obteniendo así duplicidad de información. Sin información adicional que permita saber qué porción de la onda debe ir a un canal u otro con mayor peso, la división de la onda en los canales ha de ser equitativa, por tanto, salen dos ondas iguales.
- Mayor posibilidad de atenuación de ruido. Si la grabación ha sido realizada en estéreo, quiere decir que el micrófono con el que ha sido grabado permite distinguir si la señal le llega con mayor intensidad por la izquierda o por la derecha. Con estos micrófonos, cuando hay un ruido concreto (obviando el ruido ambiente) es mucho más probable que lo detecte con más fuerza por un canal que por otro. Para que fuera equitativo, debería producirse el ruido justo delante del micrófono, para que la información en ambos canales fuera idéntica, pero esta posición la ocupa el locutor. Por tanto, de producirse el ruido, llegaría con más fuerza por un canal que por otro, mientras que al realizar la conversión a mono con la media de los valores de cada canal, el ruido se atenuará bastante conservando la voz prácticamente idéntica.

Por todo lo anteriormente descrito, una vez recortado el audio, si está en estéreo entonces se comprime en mono.

3. Cada archivo es exportado en formato WAV (audio sin compresión) y con el formato de nombre EEE_XX_NNN.wav, donde:

   a) EEE son las tres primeras letras del nombre de la emoción. Así, para alegría será Ale, para enfado, Enf, etc.

   b) XX es el número con dos dígitos (si tiene un dígito, se le añade un 0 delante) correspondiente a la frase de esa emoción, tal y como se ve en el Anexo I.

   c) NNN son las iniciales en mayúsculas del nombre de la persona que ha grabado el audio.

4. Una vez separados los audios, se agrupan por carpetas contenedoras de todos los audios

de todas las personas correspondientes a una misma emoción.

De esta manera, con las 12 personas utilizadas para esto, se consigue un total de 1.680 audios (140 audios cada persona) y con 240 audios de cada emoción.

Capítulo 3

Entorno de pruebas

Antes de iniciar los experimentos, es necesario definir qué entorno se realizará el trabajo, esto se refiere al sistema operativo en el que se va a trabajar, lenguaje de programación y su versión, librerías y ajustes generales que serán necesarios en los experimentos y que tienen todos en común. Por tanto, desarrollaremos todo lo anteriormente comentado a continuación.

3.1. Lenguajes y librerías

Para elegir el lenguaje de programación hay que atender a qué lenguajes son los más usados para estas tareas y cómo de desarrollados están; es decir, la cantidad de herramientas de las que disponen para trabajar estos campos.

- **Procesamiento de audio:** Existen una gran cantidad de lenguajes de programación especializados únicamente en el procesamiento del audio, como ChucK o Faust. Sin embargo, están orientados más al tratamiento en sí de las señales digitales que a la extracción de características. Además, al ser lenguajes dedicados, es más difícil su integración con otros lenguajes para posteriormente la construcción de la red neuronal.

  No obstante, en cuanto a librerías de lenguajes más comunes, destacan C++ y Python. Con C++ destaca la librería portaudio y con Python la librería librosa. Ambas muy completas y con una gran cantidad de funciones implementadas para extraer casi todos los tipos de características existentes en cuanto al procesamiento digital de audio.

- **Redes neuronales:** Los 3 lenguajes de programación más usados para Machine y Deep Learning, son Python, C++ y JavaScript. Partiendo de la selección previa, Python y C++ son los más potentes. C++ permite una mayor versatilidad dado que permite una programación a más bajo nivel, además, se pueden implementar algoritmos de Machine Learning más sofisticados y eficientes. No obstante, Python, con librerías como sklearn, no se queda atrás, pues tienen implementados ya todos los algoritmos de la manera más eficiente, permitiendo centrarse en el diseño de la red y trabajar a más alto nivel.

Es por las razones anteriores que la decisión está entre C++ y Python. Por familiaridad y soltura con el lenguaje en los puntos anteriores, el lenguaje escogido para el proyecto es Python, en su versión 3.7. La elección ha sido sencilla, pues es un lenguaje bastante completo, con una gran cantidad de librerías y que se adapta perfectamente a las necesidades del sistema a desarrollar.

En cuanto a las librerías, tras un breve estudio de las más completas y las que mejor se adaptan, se utilizan las siguientes:

- **librosa**[1]**:** Módulo para Python para el análisis de música y audio. Proporciona herramientas para una gran cantidad de extracción de características, transformar el audio y crear sistemas de recuperación de información musical. Esta librería es bastante completa y permite realizar casi cualquier operación definida ya sobre el audio, como por ejemplo la Transformada de Fourier (FFT).

- **scipy**[2]**:** Librería principalmente compuesta por herramientas y algoritmos matemáticos. Permite el procesamiento de señales e imágenes, interpolación, funciones especiales, FFT, etc. No tan completa como la anterior, pues librosa está centrada en el audio pero se considera muy polivalente.

- **sklearn**[3]**:** Librería de Python que proporciona herramientas para el análisis predictivo de datos y aplicable en numerosos contextos. Muy usado para clasificación, regresión,

[1] https://librosa.org/librosa/
[2] https://www.scipy.org/
[3] https://scikit-learn.org/stable/index.html

reducción de dimensionalidad, preprocesamiento, etc. Será utilizada junto con Keras para la construcción de la red neuronal.

- **Keras**[4]**:** Uno de los módulos más famosos de Python para el Deep Learning. Se basa en el concepto de una API y reduce la implementación de redes neuronales y funciones a llamadas a métodos de la API que construyen el modelo. Aun siendo una programación muy embebida de cara al programador, permite modificar prácticamente todos los parámetros del objeto o modelo que se construya, pero simplificando el proceso.

Una vez definidos el lenguaje y los módulos a utilizar, pasamos a definir otros aspectos tenidos en cuenta para el desarrollo del sistema.

3.2. Condiciones adicionales

A continuación se tratarán aquellos aspectos relacionados con las condiciones comunes a los experimentos y generales del entorno de pruebas, uno de estos aspectos es el sistema operativo. Se ha trabajado desde las versiones tempranas del proyecto y hasta prácticamente el final con el sistema operativo Ubuntu 19.10 y Ubuntu 20.04, en el que se hacían pruebas locales. Posteriormente, en los últimos experimentos, debido a que el tiempo de entrenamiento era bastante elevado, se optó por trasladar la ejecución a un fichero de Google Colab[5], utilizando la misma versión de Python y las librerías en todo momento.

En todos los experimentos, para realizar un entrenamiento y testeo del sistema, se ha seguido la técnica de **k-fold cross validation**. Esta técnica consiste en dividir el dataset en k particiones y realizar, de forma iterativa, el entrenamiento del sistema usando una partición como test y el resto como valores de entrenamiento. Finalmente, la precisión y métricas se realizan sobre la media de los resultados de cada una de las iteraciones. En todos los experimentos se han utilizado 10 particiones (10-fold cross validation) fijas.

[4]https://keras.io/
[5]https://colab.research.google.com/

Para garantizar que las particiones eran equilibradas, en lugar de recurrir a un método de alguna librería que lo realice al azar, se ha generado un script que partiendo del dataset original, distribuye los audios en 10 nuevas carpetas cumpliendo que en cada carpeta haya siemper el mismo número de audios para cada emoción. Así pues, cada partición contiene 168 archivos de los cuales cada emoción tiene 24 audios. De esta manera, se garantiza que el sistema reciba la misma cantidad de archivos de cada emoción para entrenar y no tenga ninguna emoción sobreentrenada o apenas entrenada consiguiendo así un sistema más homogéneo.

Capítulo 4

Experimento 1 (Shazam)

Habiendo definido todas las condiciones del entorno en el que se ejecutarán las pruebas, en este capítulo se explica el primer experimento, con la primera hipótesis con la que se comenzó a desarrollar el proyecto: la adaptación y aplicación del algoritmo de Shazam. En el capítulo se explicará en qué consiste, cómo se ha adaptado al problema presente y los resultados obtenidos.

4.1. Marco teórico

El primer experimento consiste en adaptar el algoritmo de la aplicación Shazam [25] al reconocimiento de emociones. Este algoritmo, considerado el más eficiente de su clase en cuanto a búsqueda, es capaz de identificar una canción con tan sólo unos segundos de muestra, sean de la parte de la canción que sea.

Para ello, se obtiene el **espectrograma** del audio y selecciona una serie de frecuencias características del audio. Un espectrograma es una gráfica tridimensional que representa la energía del contenido frecuencial de la señal según va variando a lo largo del tiempo. Cada una de las dimensiones son tiempo, frecuencia y amplitud [26]. Estas frecuencias las relaciona con otras cercanas mediante una marca de tiempo conforme a la tomada de referencia. Así mapea el audio entero repitiendo la operación, guardando todas esas frecuencias y marcas de tiempo. De esta manera, se garantiza la identificación de la canción, sea cual sea la muestra

usada.

Cuando ha realizado el mapeo del audio, con todas estas características, realiza un hash que almacena en una estructura de árbol. Cuando se le añada una muestra, lo único que tiene que hacer es recuperar de la muestra las características siguiendo el proceso anterior y buscar en el árbol la canción en cuestión. Este proceso se muestra en la figura 4.1, extraída del artículo de Wang (2003) [25].

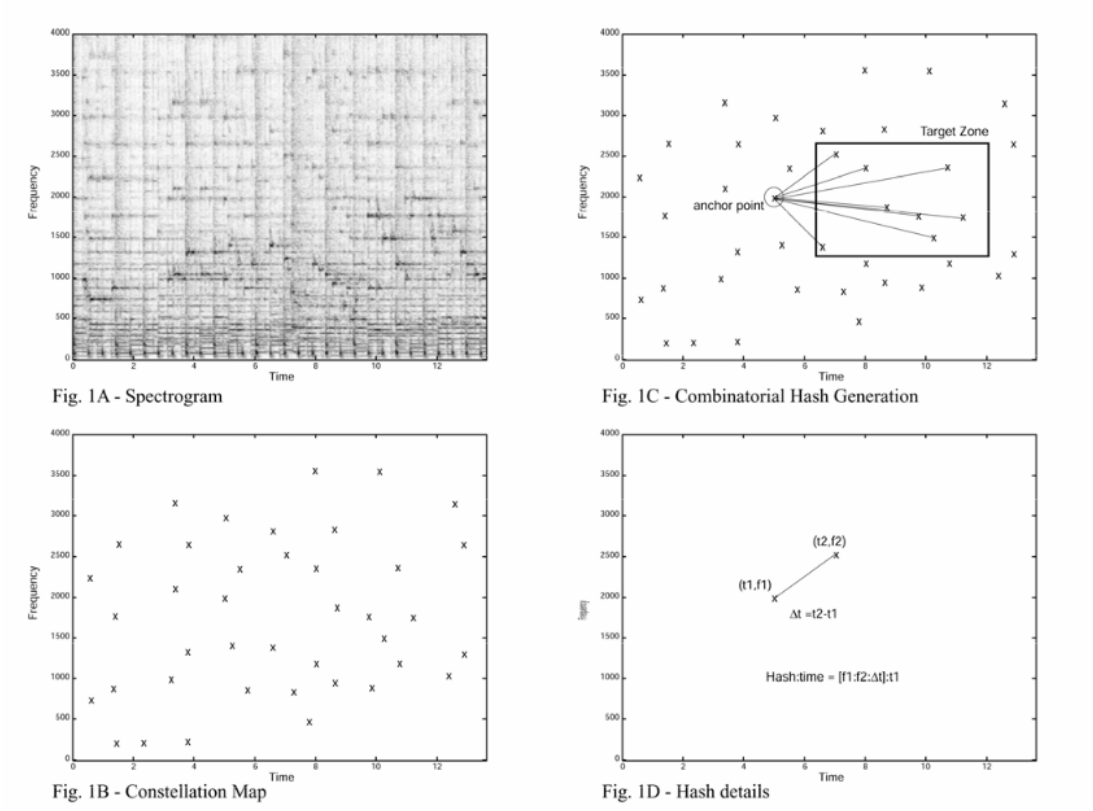

Figura 4.1: Proceso de extracción de características del algoritmo

La hipótesis que se plantea es modificar la parte del algoritmo relativo a la extracción de características para aplicarlo al caso de estudio y poder reconocer características comunes en vez de canciones. No se puede aplicar el algoritmo tal cual, pues lo único que nos daría como resultado sería la identificación de cada uno de los audios del corpus.

4.2. Características extraídas

Los principales aspectos que hay que tener en cuenta y modificar del algoritmo para adaptarlo a nuestro caso de estudio son:

- **Marca de tiempo:** El principal problema de la base de datos es la longitud de los audios. Debido a que las frases son de longitud variable y que se tiene variedad de la población en la base de datos, no todos los audios tienen la misma longitud. Por tanto, en un audio de 2 segundos para una frase, las marcas de tiempo entre frecuencias relevantes pueden ser de 100 ms, mientras que en otro de 3 segundos para la misma frase, las marcas de tiempo pueden ser de 150 ms, lo que trastocaría el algoritmo completamente.

  Esto se puede solucionar mediante porcentajes de la marca de tiempo sobre el tiempo total del audio. Así pues, para el caso anterior, en el primer audio se tendría una marca de tiempo de 0.05 (0.1 / 2 segundos) y para el segundo también de 0.05 (0.15 / 3 segundos).

- **Frecuencias:** Para escoger las frecuencias, se toman tal cual son extraídas del audio desde el espectrograma. Gracias al espectrograma, se pueden escoger las frecuencias limpiamente sin que se vean afectadas por el volumen. El volumen podría afectar en cuanto a la saturación del audio si es muy alto o por el contrario, apenas audible. El hecho de escoger las frecuencias directamente nos permite contemplar las voces agudas y graves (frecuencias más altas y más bajas) y así agrupar estas características para una misma emoción, aumentando su ratio.

Por tanto, se toma el algoritmo modificando únicamente la parte de la marca de tiempo como se ha descrito y desechando la generación del hash. Así se obtiene un vector de valores que será usado para entrenar el sistema.

4.3. Modelo y ejecución

Para llevar a cabo esta experimentación, ls red neuronal probada se describe a continuación:

- **Feed Forward Neural Network (FFNN):** También conocidas como perceptrón, fue una de las primeras redes neuronales en aparecer, descritas por Rosenblatt en 1958 por primera vez [27]. Consideradas las más básicas, surgieron como la solución a emular el comportamiento de un sistema nervioso hipotético. Constan de una capa de entrada, una o más capas (perceptrón multicapa ó multi-layer perceptron, MLP) de neuronas y una última capa de salida. Normalmente estas capas están conectadas todas con todas y realizan "feedback" entre ellas. En otras palabras, durante la fase de entrenamiento el ajuste de los pesos de cada neurona se realiza conforme a las salidas obtenidas y las esperadas. Primero se pasa la información del input por todas las capas y una vez llegado al final, se corrigen los pesos de las neuronas en orden de capa inverso, según la diferencia obtenida con la salida. Este proceso varía según el algoritmo de entrenamiento, utilizando una función matemática de reajuste u otra; no obstante el "feedback" se realiza obteniendo un valor numérico entre el estado esperado y el obtenido, y sustrayéndolo a cada uno de los pesos de las neuronas.

Se realizaron varias versiones, con diferentes números de neuronas (de 30 a 80) y diferente número de capas de neuronas (capas ocultas o hidden layers HL), que fueron de 1 a 4. El algoritmo empleado es el algoritmo BackPropagation, utilizado en este tipo de redes mayoritariamente. El vector pasado como argumento tenía 40 elementos (40 marcas de tiempo por audio). Como se ha dicho anteriormente, cada ejecución se realizó 10 veces, una con cada partición.

No obstante, los resultados fueron bastante insuficientes, no llegando a conseguir bajar el error lo suficientemente. El valor obtenido (Precision) es el mismo para todas las ejecuciones, tal y como se ve en la tabla 4.1:

4.4. Análisis de resultados

Cabe destacar, que en ninguna de las ejecuciones anteriormente mostradas se realiza un correcto entrenamiento. En todos los casos, se utiliza el error cuadrático medio para

	1 HL	2 HL	3 HL	4 HL
30 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
40 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
50 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
60 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
70 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
80 neuronas	0,1428571	0,1428571	0,1428571	0,1428571

Tabla 4.1: Resultados de ejecuciones en tanto por uno.

determinar si el entrenamiento está bien y no se sobre-entrena; cuando decae por debajo de cierto umbral, se detiene el entrenamiento. En estas ejecuciones, el error al finalizar las epochs oscilaba entre 80 y 30 (en tanto por uno), lo cual ya no lleva a considerar que los resultados obtenidos no fueron los esperados.

El hecho de obtener siempre el mismo resultado, indica que está clasificando todos los audios en una misma categoría, una misma emoción. Esto quiere decir que, tal y como mostraron Minsky y Papert en su artículo [28], los perceptrones simples, o de una capa, no pueden resolver problemas no lineales. Además sostienen que los MLP podrían resolver algunos problemas no lineales.

Viendo el planteamiento de este problema, nos está ofreciendo una idea de que resolución del problema no lineal, por lo que es necesario cambiar el enfoque y ver los posibles fallos.

4.5. Conclusiones y reajustes

Tras analizarlo en profundidad, el principal problema aparece en la extracción de características. En cuanto a las marcas de tiempo, el resultado de normalizar la marca de tiempo a un porcentaje para igualarlo en base a la duración del audio no es válido, como tampoco lo es el enfoque. Las frecuencias escogidas por el algoritmo no son consecutivas, sino que se escogen en base a las que pueden ser más relevantes para la tomada como referencia. Por tanto, para una misma frase de una emoción concreta, dependiendo de la persona, las frecuencias y tiempos tomados pueden variar enormemente. Estas variaciones pueden deberse

a la velocidad con la que hable la persona, la entonación que ésta le de (como se ve en el capítulo 2), etc.

Por tanto, es necesario desechar estas características y buscar un nuevo proceso que cumpla que:

- Las características han de extraerse sobre las frecuencias consecutivamente.
- No se pueden comparar frecuencias que disten mucho para hacer una síntesis global, estas pueden servir como identificador pero no para el caso de estudio que nos ocupa.

Capítulo 5

Experimento 2 (Waveform Contour)

Tras analizar las conclusiones obtenidas de analizar los fallos del capítulo anterior, se toma otro enfoque utilizando otras características para representar el audio. En el presente capítulo, se planteará la extracción de características del audio en base a la forma de la onda de dicho audio, y se realizarán pruebas con la red neuronal y arquitectura del capítulo anterior.

5.1. Marco teórico

Para el siguiente experimento, se pretende transformar el contorno de la señal (wave contour) a valores numéricos representativos, utilizando estos como las características identificativas de los cambios producidos en el audio. La idea parte de procesar los audios visualmente con Audacity, reconociendo algunas diferencias, a simple vista, entre los contornos de onda de unas emociones y otras.

Como se ha explicado anteriormente, cada emoción tiene una entonación distinta: la expresión de la tristeza suele producir un decaimiento de la energía, la expresión de la sorpresa suele ser muy enfatizada y homogénea, etc. Estas variaciones se pueden ver reflejadas en la forma de la onda (waveform), produciendo un tipo u otro.

La hipótesis que se propone es la siguiente: si todas las ondas de sonido tienen una forma

concreta al expresar una emoción, es posible extraer características generales de la forma de esta onda. Esto permitiría:

- No depender de las frecuencias, pues únicamente se tiene en cuenta el "dibujo" que hace la onda, el contorno.
- Obtener características de la persona que produce dicha emoción independientes del género, edad, ámbito social, etc.
- Ser robusto frente a las variaciones de velocidad del habla o la longitud del audio. El contorno de la onda será el mismo para todos los casos, solo que más o menos elongado.

Por tanto, las características que se pretenden extraer son una representación del contorno de la onda.

5.2. Características extraídas

Para explicar cómo se extrae la representación del contorno, en primer lugar es necesario definir qué se entiende como contorno de una onda. El contorno de la onda es la unión de puntos máximos relativos de la onda, tal y como se muestra en la siguiente figura 5.1.

Los puntos resaltados en rojo en la figura 5.1 son los que definen la variación de energía y frecuencias de la onda en toda su totalidad. Por tanto, las características que definen la onda son un vector de pendientes, o gradientes. Estos gradientes son la pendiente de las rectas que unen dichos puntos, es decir, la inclinación de las mismas. A una mayor variación de energía y frecuencia, mayor será la inclinación y por tanto la pendiente. Además se diferencia entre aumento y disminución: cuando la variación es de un valor menor a uno mayor, el gradiente tendrá un valor positivo, mientras que si es de un valor mayor a uno menor, el gradiente será negativo. Así pues, se construirá un vector de valores positivos y negativos que definirán la forma y el contorno de la onda. Cuanto mayor sea el número de puntos máximos que se obtengan (y por tanto de gradientes), más definido estará el contorno, y viceversa.

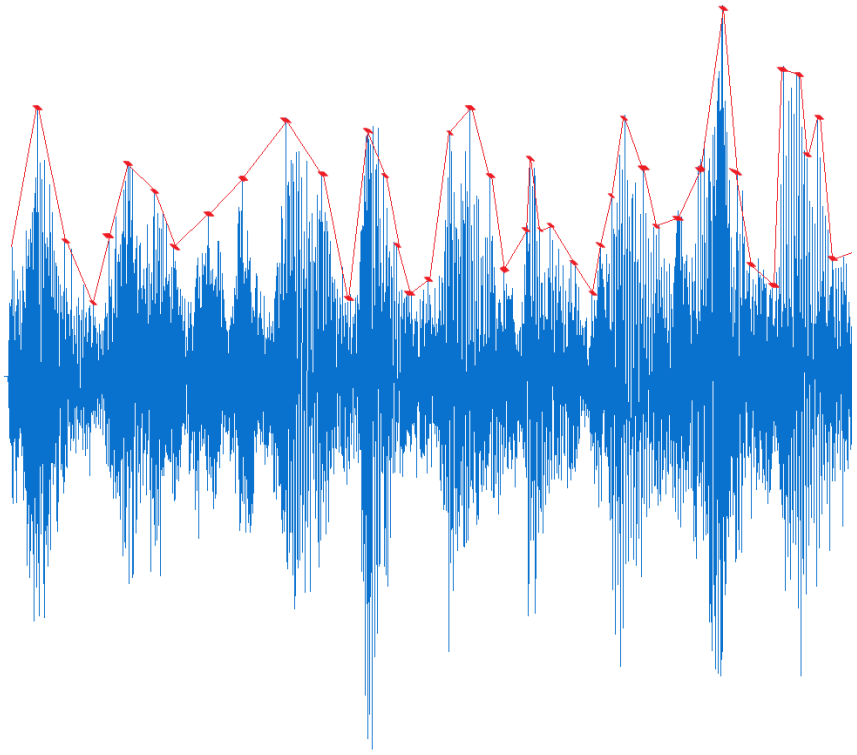

Figura 5.1: Representación de los puntos característicos de la señal. Elaboración propia.

Para el proceso, se utiliza la librería librosa, que al cargar el archivo de audio, nos devuelve dos objetos:

- **Sample rate:** El número de muestras que se toman por segundo. Esto proviene del funcionamiento del micrófono, un micrófono contiene una membrana que vibra al recibir una onda de sonido y realiza tantas mediciones del estado de la membrana por segundo, como valor tenga el sample rate. Así pues, para un sample rate de 22.050 Hz, se miden y guardan 22.050 veces por segundo el estado de la membrana del micrófono. A mayor sample rate, se consigue mayor nitidez del sonido y viceversa.

  Este valor depende de los medios utilizados en la grabación, por lo que puede variar de un audio a otro de la base de datos. Además, no se puede modificar este valor puesto que, si se cambia a un valor menor que el nativo del audio, se pierde información, por el contrario, si se amplía a uno mayor faltaría información para completar el total de

muestras. No obstante, esto puede suplirse fácilmente con la librería, pues devuelve el valor nativo del audio en cuestión.

Este valor es útil para saber, entre otras cosas, la duración del audio exacta. Teniendo el número total de muestras y cuántas se han tomado por segundo (sample rate), se puede obtener la duración.

- **Muestras:** Nos devuelve la serie temporal de audio del archivo. Esta serie está compuesta por valores medidos y ordenados cronológicamente, conteniendo varios tipos de información en estos valores. En el caso del audio, las principales características del sonido que van unificadas son intensidad y altura, volumen (energía) y frecuencias. Para la definición de contorno dada, se trabaja con estos valores pues ya vienen combinados. Si se separasen y se trabajara con alguno de ellos por separado, se perdería la información relativa al campo desechado. Una vez obtenidas las muestras, se procede a producir el vector de gradientes:

  1. **Normalización de muestras.** En primer lugar se normalizan las muestras al intervalo [-3.500, 3.500]. Esto se hace para fijar un gradiente máximo y mínimo y trabajar con un rango definido de valores. De otra forma, el gradiente máximo podría ser un valor que diste mucho de los demás, por ejemplo, si existiera algún pico de energía se podría tomar como relevante cuando en realidad no lo es.
  2. **Frecuencias máximas relativas.** Una vez normalizadas las muestras se recorre en un vector almacenando los valores máximos relativos positivos, esto es, guardar aquellos valores que sean mayores de 0 y que a su vez sean mayores que su inmediato anterior y su inmediato posterior. En el vector de puntos se almacena, en una tupla, su posición en el vector de muestras y el valor en cuestión.

  $$x_i > 0, \quad x_i > x_{i\ 1}, \quad x_i > x_{i+1} \quad \rightarrow \quad (i, x_i)$$

  3. **Reducción del vector.** Una vez se tiene el vector, se procede a su reducción. Si se pretende construir un vector final de N gradientes, se deberá reducir a N + 1

puntos (ya que los gradientes se calculan entre dos puntos). Para reducir el vector se realizan las siguientes operaciones:

a) Calcular la distancia geométrica entre cada punto almacenado y el siguiente, y se guarda en un vector auxiliar, almacenando la posición del punto y la distancia con su consecutivo.

b) Se ordena de mayor a menor distancia (se da prioridad, o más peso, a los puntos que tengan una mayor diferencia de valores de muestras).

c) Se trunca el vector manteniendo el 75 % inicial de los puntos (o con N + 1 puntos si al truncar quedan menos de N + 1 puntos).

d) Se eliminan del vector inicial todos aquellos puntos que no hayan pasado el corte.

e) Repetir operación hasta llegar a N + 1 puntos.

4. **Conversión a gradientes.** Con el vector reducido, se calcula el gradiente para cada punto con el siguiente (a excepción del último punto), obteniendo así el vector de gradientes de N puntos. El gradiente se calcula mediante la siguiente fórmula:

$$\forall i < N \quad \rightarrow \quad gradient_i = (x_{i+1,2} - x_{i,2})/(x_{i+1,1} - x_{i,1})$$

5.3. Modelo y ejecución

Para las pruebas con estas características, se utiliza una red neuronal con la misma configuración que en el experimento anterior: un perceptrón multicapa con distintas configuraciones. Probadas de 1 a 4 hidden layers y variando el números de neuronas entre 20, 40, 50, 60 y 75. No obstante, se hicieron pruebas con vectores de distinto tamaño N. Las pruebas realizadas se hicieron con los valores de N 50, 75, 80 y 100. Los resultados de las ejecuciones fueron los siguientes:

N=50	1 HL	2 HL	3 HL	4 HL
20 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
40 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
50 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
60 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
75 neuronas	0,1428571	0,1428571	0,1428571	0,1428571

Tabla 5.1: Waveform contour. Resultados de ejecuciones con N=50 en tanto por uno.

N=75	1 HL	2 HL	3 HL	4 HL
20 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
40 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
50 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
60 neuronas	0,1428571	0,1428571	0,1428571	0,0101190
75 neuronas	0,1428571	0,1428571	0,0101190	0,0101190

Tabla 5.2: Waveform contour. Resultados de ejecuciones con N=75 en tanto por uno.

N=80	1 HL	2 HL	3 HL	4 HL
20 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
40 neuronas	0,1428571	0,1428571	0,1428571	0,1428571
50 neuronas	0,1428571	0,1428571	0,0101190	0,0101190
60 neuronas	0,1428571	0,0101190	0,0107142	0,0113095
75 neuronas	0,1428571	0,0107142	0,0107142	0,0113095

Tabla 5.3: Waveform contour. Resultados de ejecuciones con N=80 en tanto por uno.

N=100	1 HL	2 HL	3 HL	4 HL
30 neuronas	0,1428571	0,1428571	0,0101190	0,0125000
40 neuronas	0,1428571	0,1428571	0,0130952	0,0130952
50 neuronas	0,1428571	0,0107142	0,0148809	0,0184523
60 neuronas	0,0107142	0,0119047	0,0178571	0,0190476
70 neuronas	0,0130952	0,0125000	0,0190476	0,0202381
80 neuronas	0,0125000	0,0125000	0,0196428	0,0202381

Tabla 5.4: Waveform contour. Resultados de ejecuciones con N=100 en tanto por uno.

5.4. Análisis de resultados

Como se puede observar, se consigue una ligera mejora en comparación con los resultados del experimento anterior. No obstante, en la inmensa mayoría de ejecuciones sigue sin entrenar lo suficiente (volviendo a clasificar todos los audios en la misma emoción) y no es hasta que se prueba con vector de 80 elementos (en la tabla **??**), que se consigue entrenar mínimamente.

Aun así, los resultados siguen siendo muy pobres, aunque podría probarse que, con más neuronas y más capas, se llegaría a conseguir una solución decente. El problema de este planteamiento es que se deja la solución en la base de "con recursos infinitos, se llega a la solución". Antes de ver si la solución es dar recursos infinitos, es revisar el planteamiento para ver si es coherente o no.

Tras un análisis, se llega a la conclusión de que las características utilizadas no sirven por los siguientes motivos:

- Se está tratando con una serie temporal, en la que un valor está compuesto por otros a su vez. Si se trata cada valor de la serie como la suma de intensidad y altura (simplificando mucho), un mismo valor puede conseguirse con varias combinaciones de intensidad y altura. Esto supondría que, por ejemplo, un audio grabado con voz grave y hablando fuerte pudiera equipararse a otro audio grabado con voz aguda y menos volumen. Como se ha dicho anteriormente, el volumen no puede ser obviado, pues contiene información relevante para la identificación de una emoción.

- Si se aplica la transformada de Fourier para aplicar el mismo procedimiento a las frecuencias (sin el añadido de la energía, tal y como se obtenían en el experimento anterior), se encuentra un problema de base. El gradiente se realizaría únicamente entre las frecuencias de un audio y la diferencia de frecuencias.

  Como se ha mencionado en la introducción de este trabajo, las voces graves tienen frecuencias más bajas, mientras que las voces más agudas tienen frecuencias más altas. De este modo, una voz grave se moverá en un rango de frecuencias más reducido que una voz aguda, y por tanto, las pendientes serán siempre menos pronunciadas que una voz aguda. De esta manera, aunque dos personas con voces aguda y grave digan la misma frase, a la misma velocidad y realizando las mismas variaciones en la entonación, aquella persona que tenga la voz más grave tendrá valores de gradiente menores que la otra.

- Por otro lado, si para el caso anterior se aplica una normalización de las frecuencias, equiparas los audios que tengan frecuencias más bajas con otros que las tengan más agudas, perdiendo la diferencia que los separaba. Esta normalización implica escalar todas las frecuencias a un rango determinado, forzando a aumentar las frecuencias graves y/o a disminuir las agudas; lo que resulta en una distorsión de la onda que no representa a una voz real.

5.5. Conclusiones y reajustes

Vistos los problemas de este enfoque, queda de manifiesto que es necesario utilizar alguna alternativa que permita separar estas series temporales y trabajar con todas las cualidades del sonido. Una solución inicial posible es la Transformada de Fourier, pues nos descompone los valores de la serie en otros que se pueden manejar mejor.

No obstante, hay que tener en cuenta el problema de la diferencia de frecuencias en voces graves y agudas. La explicación más sencilla a este problema viene de la rama de la acústica en teoría música. Cada nota musical tiene una frecuencia concreta asociada; es el caso de la nota "la", y más concretamente la4 o A4 (en notación internacional). Esta nota, utilizada como referencia para afinar instrumentos, es el "la" central en el piano, y tiene una frecuencia de 440 Hz.

Sin embargo, hay más notas "la" en el piano. Estas notas de igual nombre, se llaman octavas y cumplen la regla de dos sonidos son octavados si cumplen una relación de frecuencia fundamental de dos a uno. En otras palabras, para el inmediato "la" más bajo (la3), su frecuencia será la mitad de la4, 220 Hz. Y así para la5 que tendrá 880 Hz, etc.

Entre dos notas octavadas hay 11 notas; entre l3 y la4 hay 11 notas, entre la4 y la5 hay 11 notas, y así sucesivamente; sin embargo, al ser más graves o agudas , se reparten en un rango de frecuencias distinto. Así, para el rango de la3 y la4, hay 11 notas repartidas en un rango de 220

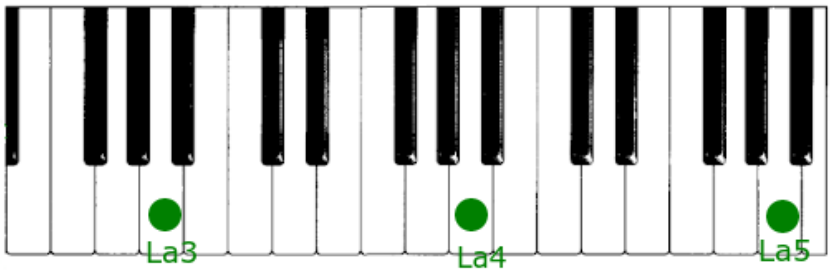

Figura 5.2: Ejemplo de distribución de notas en un piano. Elaboración propia.

Hz [220, 440], para el rango de la3 y la4 hay un rango de 440 Hz [440, 880] y así sucesivamente.

Para conseguir medir bien la entonación, es necesario algún sistema que no se base únicamente en el valor de las frecuencias, sino que pueda transformar esos rangos a "saltos". Con el ejemplo anterior, es necesaria alguna técnica que en vez de medir que de la3 a la4 hay un salto de 220 Hz, hay 11 notas y , de igual manera, que de la4 a la5 hay 11 notas, en vez de 440 Hz de diferencia. De este modo podría decirse que hay la misma distancia de la3 a la4 que de la4 a la5.

Por tanto, se llega a la conclusión de que para los siguiente experimentos, las características deben cumplir esta última conversión y que además utilicen los valores descompuestos de las series temporales.

Capítulo 6

Experimento 3 (MFCCs)

Teniendo en cuenta que las características deben ser mucho más complejas, tratando energía, frecuencia, etc. en el capítulo a continuación, se realiza un nuevo enfoque extrayendo un nuevo tipo de características más representativas de los cambios en el audio, llamados los coeficientes de Mel. Dichas características se estudian para adaptarlas al caso de estudio y se prueban en redes neuronales de tipo LSTM y CNN.

6.1. Marco teórico

Tras las conclusiones extraídas del experimento anterior, es necesario realizar un estudio de técnicas de procesamiento de audio y relacionadas con las disciplinas que se nombraron inicialmente: reconocimiento del locutor, análisis de voz, etc. Tras realizar un análisis bibliográfico, se opta por la técnica de los Coeficientes Cepstrales en las Frecuencias de Mel (MFCCs).

6.1.1. Coeficientes Cepstrales en las Frecuencias de Mel

Para comprender bien lo que se va a realizará en este experimento, es necesario conocer qué son los coeficientes, cómo funcionan y cómo pueden ser aplicados al reconocimiento de emociones. Los coeficientes son una representación del habla bajo la percepción auditiva humana. El mecanismo que se emplea para obtener estos coeficientes es muy similar al

funcionamiento del oído humano.

Técnicamente, los coeficientes son una representación del espectrograma de energía en pequeños tramos basados en la transformada linear del coseno sobre el espectrograma obtenido de la escala de frecuencias de Mel. A continuación, se explica la obtención de estos coeficientes, los cuales nos llevará a una mejor comprensión de la cuestión:

1. **Framing.** Este proceso inicial consiste en separar la señal en pequeños tramos. Esto se debe a que cualquier señal de audio (tomada en condiciones normales) está cambiando constantemente en el tiempo por muchos factores, lo que hace complicado obtener características relevantes. Al variar tanto en el tiempo, las características importantes pueden pasar desapercibidas si se procesa la señal completa. Por este motivo, se divide en ventanas en las que la señal varía poco de manera que se puedan extraer características invariantes para cada tramo de la señal.

   Para esto, se separa en ventanas de 2.048 muestras con 512 muestras entre tramo y tramo. El total de características en conjunto representan la señal total. Estos valores son extraídos mediante experimentación, de forma que son los que mejor funcionan a la hora de emular el oído humano mediante este procesamiento.

2. **Windowing.** Una vez se ha dividido la señal en tramos, se le aplica una función de windowing (a cada tramo) Hamming window, por defecto. Una función de windowing consistse en una función matemática que "suaviza" el tramo de señal pasado para evitar la discontinuidad al principio y al final del bloque analizado; le otorga coherencia al bloque.

   Estas funciones se usan cuando el análisis de una señal se realiza sobre una señal deliberadamente limitada (como los tramos en los que se ha dividido la señal). La de Hamming es la más utilizada en este caso. Otro motivo por el que se hace es para contrarrestar el principio que asume la Transformada de Fourier de que se aplica sobre una señal infinita. Esto permitirá obtener características más fiables de la FFT.

3. **Transformada de Fourier.** El siguiente paso es aplicar la Transformada de Fourier a cada uno de los tramos para obtener todas las ondas que componen la original del tramo, devolviendo la señal transformada en tiempo o espacio y el dominio de la frecuencia. De esta forma, se extraen todas las frecuencias originarias de dicho tramo para poder procesarlas posteriormente. De la FFT se obtiene el espectro de energías de todos los tramos de la señal.

4. **Banco de filtros de la escala de Mel.** Una vez realizada la Transformada de Fourier y calculado el espectro de energías, el siguiente paso es aplicar los filtros de Mel para transformar las frecuencias a la escala de Mel.

   La escala de Mel es una escala musical perceptual de tonos tratados como intervalos equiespaciados por parte de los observadores. Como se decía en las conclusiones del experimento anterior, las notas musicales van doblando su frecuencia conforme van octavándose. Esto refleja un crecimiento exponencial de las frecuencias, conforme más agudas son las notas.

   La escala de Mel transforma dichas distancias en frecuencias (de forma logarítmica para contrarrestar el crecimiento exponencial) obteniendo un crecimiento lineal a partir de los 500 Hz. A partir de los 500 Hz, cuatro octavas en una escala musical normal, se verían comprimidas a dos escalas. Gracias a esto, se pueden tratar los "saltos"de frecuencias como lineales; como se concluía antes, de la3 a la4 hay 11 notas y de la4 a la5 hay 11 notas.

   De esta manera, se convierten los valores del espectro de energías obtenido a la escala de Mel mediante solapamiento de ventanas triangulares, consiguiendo así las energías de cada uno de los tramos de la señal. Las frecuencias que aquí se obtienen son muy similares a las que recibe un ser humano al oír la misma señal.

5. **Logaritmos de las energías.** Aunque el rango de frecuencias obtenido es muy similar al del ser humano, la diferencia es que el ser humano no percibe el crecimiento de los sonidos linealmente, sino logarítmicamente. Por tanto, se le aplica el logaritmo a estas energías de Mel.

6. **Transformada del Coseno Directa.** En último lugar, se le aplica la Transformada Directa del Coseno (DCT) para conseguir el inverso de la transformación inicial (con la FFT) y obtener la señal original con todos los cambios aplicados para que sea lo más similar posible a lo que percibe el ser humano. No obstante, la señal sigue estando dividida en tramos.

   Si se obtiene el espectro de esta señal, las amplitudes de dicho espectro son los coeficientes cepstrales de Mel.

Estos coeficientes, son una representación adecuada, tanto de las características estáticas de cada uno de los tramos de la señal, como de las dinámicas en ellos para la correcta detección e interpretación del sonido. Sin embargo, representan únicamente el espectro de poder envolvente de cada tramo, por lo que se pierde información dinámica entre dichos tramos. Por tanto se calculan y añaden unos valores llamados Deltas y Delta-Deltas. Estos valores representan la velocidad (deltas) a la que cambian los coeficientes entre los tramos en los que se dividió la señal, y la aceleración (delta-deltas). Estos valores pueden ser o no relevantes, según el caso. Para el reconocimiento del habla, Speech Recognition (SR) en inglés, pueden beneficiar los resultados; no obstante, para este caso no se tendrán en cuenta inicialmente. Por tanto, se explorarán más detenidamente en la siguiente sección, para decidir cómo adaptarlos al proyecto.

6.2. Características extraídas

Normalmente, los coeficientes que se utilizan en SR son los 13 primeros o los coeficientes 2-13 de todos los extraídos y calculados (suelen ser 40). Estos coeficientes son suficientes para determinar características relevantes para la dicción, el resto suele aportar ruido ya que representan cambios rápidos en los filtros de Mel. En cambio, para la detección de emociones sí pueden ser útiles ya que esa información representa las variaciones de la entonación en el habla, que es justo lo que se necesita para el sistema.

El número de coeficientes que se aplicarán son 40. No hay ningún estudio que justifique que 40 es el número óptimo para estos casos, sino que por experimentación se llega a la conclusión de que las variaciones que representan los coeficientes del 41 en adelante, no aportan información relevante. Por lo tanto, se trabajará ahora con matrices de 40 mfccs x N, siendo N el número de tramos en los que se divide la señal.

Todo este proceso es resumido y proporcionado, de manera muy simple, por la librería librosa en una única función. Únicamente necesita como parámetros el vector de datos de las series temporales, el sample rate nativo del audio y el número de coeficientes que se quieren obtener.

Al observar los resultados, es donde se aprecia el siguiente problema: al no tener todos los audios la misma longitud y dado que todos se dividen en un tamaño estático para calcular los coeficientes, cada matriz de coeficientes tiene una dimensión distinta para cada audio. Las dimensiones varían desde [40, 80] hasta [40, 1289]. Así se ve que el audio más breve de la base de datos se divide en 80 tramos y el más extenso en 1289 tramos. Esto es un inconveniente puesto que para entrenar la red neuronal, se necesita que todos los inputs o entradas sean del mismo tamaño.

La solución consiste en reducir o ampliar las matrices hasta tener una dimensión fija, pero antes de determinar las dimensiones es necesario determinar cómo va a ser esta redimensión.

Si la matriz se aumenta, hay que rellenar con valores. Normalmente, para otros tipos de características, se rellenan con valores aleatorios pero no en este caso para nuestro problema. Rellenar la matriz hasta completar el tamaño deseado con valores aleatorios, implica que la señal se comporta de manera aleatoria en estos valores, lo que se traduce al procesarlo en ruido.

El procedimiento para igualar señales de sonido en el procesamiento de audio es cortar el audio en la longitud N, desechando todo lo demás o añadiendo silencio a continuación

si la señal tiene una duración menor que la deseada. Como los coeficientes representan las amplitudes del espectro de energía, la energía cuando hay silencio es 0. Por tanto, si la matriz tiene un tamaño menor al deseado se le añaden ceros hasta completarla y si tiene un tamaño mayor se trunca para quedarse con los primeros N tramos.

Aún queda determinar el valor de N. Para obtener el valor que mejor se ajusta, y así determinar un tamaño representativo de todos los audios conformantes del corpus , se calculan los MFCCs para todos los audios y se almacena la segunda dimensión de la matriz resultante en dos vectores:

- Un vector ordenado de menor a mayor con todos los valores existentes, estén repetidos o no.
- Un vector ordenado de menor a mayor con todos los valores únicos existentes. Es decir, si hay tres audios cuyos MFCCs tienen dimensión [40, 235], el número 235 aparece una única vez en la lista.

Con estos valores almacenados, se estudian los valores correspondientes a los 3 cuartiles para cada uno:

1680 audios	Valores repetidos	Valores únicos
Primer cuartil	170	180
Segundo cuartil	205	272
Tercer cuartil	253	377

Tabla 6.1: Cuartiles de todas las dimensionalidades de los MFCCs. Elaboración propia.

El tercer cuartil de los valores repetidos quiere decir que si se ordenan los audios de menor a mayor número de tramos en los que se divide, tres cuartas partes de los audios se dividen en un número igual o inferior a 253, y análogamente con 377 para los valores únicos. Por tanto, las pruebas se realizan tomando esos valores con dimensiones de [40, 256] (para redondear a un número potencia de dos) y [40, 377], siendo así un valor representativo de la base de datos. Así se garantiza que un gran número de audios no pueda ser truncado.

Para extraer los MFCCs se crea un script que procesa los audios partición por partición almacenando los resultados en un DataFrame de la librería pandas. El proceso es el siguiente:

1. Para cada partición crea un DataFrame.

2. Para cada audio, mediante la función librosa.feature.mfcc(), se obtienen los 40 MFCCs.

```
mfccs=librosa.feature.mfcc(y=audio,sr=sample_rate,n_mfcc=40)

```

3. Redimensionar los MFCCs al tamaño en cuestión, que se denominará avg_n_mfcc ([40, 256] ó [40, 377]).

```
    if avg_n_mfcc > mfccs.shape[1]:
        elems = list(mfccs.T)
        for i in range(avg_n_mfcc - mfccs.shape[1]):
            elems.append(np.zeros(num_mfcc))
        mfccs = np.array(elems).T
    else:
        mfccs = mfccs[:,:avg_n_mfcc]

```

4. En base al nombre del archivo, se guarda la etiqueta con la emoción correspondiente, por ejemplo, Ale para Alegría y Sor para Sorpresa y así sucesivamente.

5. Se añade como lista de dos elementos al DataFrame, en el que el primer elemento son los MFCCs y el segundo la etiqueta.

6. Cuando ha procesado todos los audios de la partición, exporta el DataFrame a un archivo binario en formato DAT con el nombre ParticionX_mfcc.dat, siendo X el número de la partición en cuestión.

De esta forma, no es necesario volver a calcular los MFCCs para cada audio en cada prueba que se haga con las mismas condiciones. En el modelo simplemente habrá que cargar los datos de estos archivos binarios y ejecutar el experimento.

6.3. Modelo y ejecución

6.3.1. Redes neuronales

En este experimento, se abandona el perceptrón para centrarse en dos nuevos tipos de redes neuronales: Long short-term memory (LSTM) y Convolutional Neural Networks (CNN). Las pruebas se realizan sobre estos dos redes puesto que son los más abundantes en los artículos relacionados con estas temáticas.

- **LSTM.** Son un tipo de redes neuronales recurrentes que, a diferencia de los perceptrones tienen conexiones de feedback. Presentadas en 1997 por Hechreiter y Schmidhuber [29] como una mejora de las redes neuronales recurrentes (RNN) están compuestas por capas de celdas de memoria. Cada celda a su vez, tiene 3 puertas:

  - Input. Decide cuánta información se almacena en la celda.
  - Output. Decide cuánta información se pasa a la siguiente celda.
  - Forget. Decide qué información se almacena en la celda.

  Las puertas tienen como función salvaguardar la información permitiendo o bloqueando el flujo de información a través de ellas. Son muy buenas en clasificación, procesamiento y predicciones basadas en series de datos. Permiten aprender con datos en los que el orden importa (series temporales) gracias a que almacenan memoria.

- **CNN.** Este tipo de redes neuronales fue propuesto en 1998 por LeCun, Bottou, Bengio y Haffner [30] como una nueva alternativa. Muy diferentes al resto, contienen un escáner (especificado como `kernel_size`) que divide los inputs para analizarlos poco a poco, en otras palabras, crea una ventana de unas dimensiones concretas que aplica a los inputs.

  Estos inputs son enriquecidos a través de una serie de capas convolucionales en las que únicamente están conectados los nodos más cercanos, no todos con todos. Estas capas van siendo reducidas a números divisibles por el tamaño del input.

  En ocasiones, se añade una FFNN al final para un doble procesamiento de los datos y

una mejora en los resultados. Suelen ser muy utilizadas en el procesamiento de imágenes donde se obtienen muy buenos resultados.

La configuración inicial de ambas redes neuronales se muestra a continuación:

Para la red LSTM se construye una red con dos capas de neuronas la primera con 128 neuronas y la segunda con 32, con un dropout (cuánta información se borra) de 0.25 (un 25 % de la información), que se variará con valores de [0.1, 0.15, 0.2, 0.25] y recurrent_dropout (cuánta información se pasa a la siguiente celda) de 0.15, que variará en [0.05, 0.1, 0.15, 0.2]. El método de activación es softmax y se utiliza el optimizador Adam (que usa el método del descenso del gradiente estocástico) [31] . Para el error se utiliza categorical_crossentropy y la métrica de salida es la precisión. Por último, el sistema es entrenado con 400 epochs.

A continuación, se muestra la estructura de la red LSTM en código, con la configuración que mejores resultados proporcionaba. Más concretamente, el código muestra la construcción y configuración del modelo usado para entrenar, con las capas de neuronas, optimizador, algoritmo de entrenamiento, etc.

```
model = Sequential()

model.add(LSTM(units=128, dropout=0.25, recurrent_dropout=0.15,
return_sequences=True, input_shape=input_shape))
model.add(LSTM(units=32,  dropout=0.25, recurrent_dropout=0.15,
return_sequences=False))
model.add(Dense(units=train_Y.shape[1], activation="softmax"))

opt = Adam()
model.compile(loss="categorical_crossentropy", optimizer=opt,
metrics=["accuracy"])
model.summary()

batch_size = 35  # num of training examples per minibatch
num_epochs = 400
model.fit(
    train_X,
```

```
    train_Y,
    batch_size=batch_size,
    epochs=num_epochs,
)
```

Para la CNN en cambio se utilizan inicialmente 4 capas de convolución 2D doblando el número de filtros en cada capa y aplicando una ventana (kernel_size) de tamaño (2, 2). De esta manera, la primera tenía 16, la segunda 32 y posteriormente 64 y 128. Tras cada capa, se aplica una capa de MaxPooling2D la cual disminuye y suaviza la representación del input tomando el valor máximo sobre la ventana definida para cada dimensión a lo largo del eje de características, es decir, tras cada convolución se le aplica una ventana para disminuir los valores, esta ventana es de tamaño (2, 2). Tras esta capa de MaxPooling2D se le aplica un dropout (equivalente al learning rate) que variará en [0.1, 0.15, 0.2, 0.25]. Finalmente se entrena el sistema con 100 epochs.

A continuación se muestra la construcción y configuración del modelo con CNN, en código, que mejores resultados proporcionaba. Se muestran las 4 capas de convoluciones, la condensación en las categorías finales y los algoritmos de entrenamiento y optimización.

```
model = Sequential()
model.add(Conv2D(filters=16, kernel_size=2,
input_shape=(num_rows, num_columns, num_channels), activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.1))

model.add(Conv2D(filters=32, kernel_size=2, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.1))

model.add(Conv2D(filters=64, kernel_size=2, activation='relu'))
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.1))

model.add(Conv2D(filters=128, kernel_size=2, activation='relu'))
```

```
model.add(MaxPooling2D(pool_size=2))
model.add(Dropout(0.1))

model.add(GlobalAveragePooling2D())
model.add(Dense(num_labels, activation='softmax')

model.compile(loss='categorical_crossentropy', metrics=['accuracy'],
optimizer='adam')

num_epochs = 100
num_batch_size = 32

model.fit(x_train, y_train, batch_size=num_batch_size, epochs=num_epochs,
validation_data=(x_test, y_test), callbacks=[checkpointer], verbose=1)
```

6.3.2. Tratamiento de los audios

Los datos estaban guardados en DataFrames por particiones por lo que para realizar el 10-fold cross validation se sigue el siguiente proceso:

1. Al inicio del sistema se cargan todos los dataframes y se almacenan en una lista en orden de partición.

2. En un bucle for de 10 iteraciones, se guarda en un DataFrame la partición que se usará de test y el resto en otro que se usará de train.

3. Una vez se tienen los dataframes de train y test, se almacenan los valores de dichos dataframes en vectores de numpy, separando MFCCs y etiquetas.

4. Para las etiquetas, se utiliza un LabelEncoder para convertir los valores de string a vectores numéricos, de esta manera, para Alegría corresponderá el vector [1, 0, 0, 0, 0, 0, 0], para Asco el vector [0, 1, 0, 0, 0, 0, 0], etc.

6.3.3. Primeros resultados

Los resultados para las ejecuciones (tras el 10-fold cross validation) se muestran en la tabla 6.2 para los obtenidos con LSTM, y en la tabla 6.3 para los obtenidos con CNN.

LSTM		Dropout			
[40, 256]		0.10	0.15	0.20	0.25
recurrent dropout	0.05	0,256547	0,263095	0,266666	0,264285
	0.10	0,261309	0,268452	0,272023	0,270833
	0.15	0,269047	0,273809	0,280357	0,277976
[40, 377]		0.10	0.15	0.20	0.25
recurrent dropout	0.05	0,286309	0,290476	0,297619	0,293452
	0.10	0,289881	0,290476	0,293452	0,293452
	0.15	0,294642	0,303571	0,306547	0,304761

Tabla 6.2: Resultados ejecución LSTM.

CNN	Dropout			
	0.10	0.15	0.20	0.25
[40, 256]	0.660714	0.655357	0.648214	0.639285
[40, 377]	0.672619	0.660714	0.660119	0.655357

Tabla 6.3: Resultados ejecución CNN.

6.4. Análisis de resultados

Como se aprecia en los resultados, la red convolucional produce muchísimo mejores resultados que la red LSTM obteniendo en el mejor de los casos un 67 % de acierto. No obstante, es necesario analizar los resultados más detenidamente:

- La mejor dimensión para la matriz de los MFCCs en ambos casos es (40, 377). De aquí se extrae que se debe a que un menor número de audios son truncados, por lo que la mayoría son procesados sin ser modificados y añadiendo silencio al final de los audios hasta completar la duración total . Esto quiere decir que existe información relevante en cómo termina la entonación de la frase, por lo que es preferible procesar audios cortos y añadir 0 a la matriz de MFCCs que procesar audios muy largos y tener que truncarlos.

- De acuerdo al uso de dropout, existen discrepancias entre las dos redes neuronales, obteniendo mejores resultados la LSTM con valor 0.2, mientras que la CNN obtiene mejores resultados con valor 0.1. Sería relevante si funcionara de la misma manera, pero no hay que olvidar que tienen un funcionamiento completamente distinto, por lo que el hecho de que un valor funcione mejor para uno u otro, no es excesivamente relevante.

Como se comentaba en la sección anterior, las redes convolucionales son muy utilizadas en el procesamiento de imágenes, es por esto que inicialmente se consiguen muy buenos resultados con la matriz de coeficientes. Los valores están en unos rangos concretos y están relacionados tanto en horizontal como en vertical (en cuanto a la información que representan). Por tanto, en cierto modo, es como si se estuviera utilizando la matriz de MFCCs como una imagen representativa del audio para entrenar.

6.5. Conclusiones y reajustes

Aun habiendo conseguido buenos resultados (sobre todo en comparación a los resultados obtenidos en el experimento 2), quedaban muchas configuraciones por probar, de esta forma y con el ánimo de mejorar aún más los resultados, se probaron muchas más configuraciones de la red neuronal convolucional, hasta dar con una que superó el 70 % (0.718452) de acierto sin llegar a sobreentrenar.

A continuación se mostrará dicha configuración y, tras ella, se explicará todos aquellos campos modificados y los valores que se probaron:

```
model = Sequential()

model.add(Conv2D(filters=32, kernel_size=(3, 3),
input_shape=(num_rows, num_columns, num_channels), padding='same',
activation='relu'))
model.add(Conv2D(filters=32, kernel_size=(3, 3), padding='same',
activation='relu'))
model.add(Dropout(0.1))
```

```
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='same',
activation='relu'))
model.add(Conv2D(filters=64, kernel_size=(3, 3), padding='same',
activation='relu'))
model.add(Dropout(0.1))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(filters=128, kernel_size=(3, 3), padding='same',
activation='relu'))
model.add(Conv2D(filters=128, kernel_size=(3, 3), padding='same',
activation='relu'))
model.add(Dropout(0.1))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(GlobalAveragePooling2D())
model.add(Dense(num_labels, activation='softmax'))

model.compile(loss='categorical_crossentropy', metrics=['accuracy'],
optimizer='adamax')

num_epochs = 72
num_batch_size = 32

model.fit(x_train, y_train, batch_size=num_batch_size, epochs=num_epochs,
validation_data=(x_test, y_test), callbacks=[checkpointer], verbose=1)

_________________________________________________________________
Total params: 287,335
Trainable params: 287,335
Non-trainable params: 0
_________________________________________________________________
```

Los valores probados y modificados han sido los siguientes:

- **Número de capas.** Una de las principales modificaciones es variar el número de capas, la continuidad entre las mismas, el tamaño, etc. Se probaron configuraciones con hasta 8 capas iguales dos a dos (dos capas de 16, dos de 32, y así sucesivamente), y se encontraron dos configuraciones que mejoraban inicialmente el porcentaje conseguido. Pero al generar nuevas particiones, esas configuraciones bajaban rápidamente su porcentaje de acierto mientras que esta lo mantenía. Además es una solución más elegante que poner muchas capas de distinto tamaño y entremezclando.

- **Kernel size.** En cuanto al tamaño del kernel, la experimentación fue más rápida. El kernel indica el tamaño de la ventana que se aplica a los datos para realizar la convolución. Esto implica que una ventana de tamaño 1 ó (1, 1), apenas enriquezca los datos, pues solo tiene como valor el propio dato.

  Con tamaño 2 ó (2, 2), la convolución, al aplicarse sobre un valor x, en la ventana de convolución entrará el siguiente a este (del coeficiente que representa el siguiente tramo) y los correspondientes al siguiente coeficiente. Dicho de otro modo, se tiene en cuenta cómo avanza el audio.

  Con tamaño 3 ó (3, 3), se consigue que se tenga en cuenta cómo avanza el audio y de dónde viene, al tener en cuenta el siguiente y anterior coeficientes, y el posterior y anterior tramo del audio. Por tanto, mediante experimentación se corrobora que con tamaño (3, 3) de kernel se obtienen mejores resultados.

  También se pueden probar tamaños con 5 y 7, pero los resultados son muy parecidos a los de tamaño 2, ya que al tener en cuenta más información, la información relevante pasa más desapercibida al realizar un tratamiento más global.

- **Padding.** El padding consiste en el relleno de los datos de entrada para que tengan el mismo tamaño que los de salida. Esto es utilizado en el MaxPooling2D para antes de aplicar el "suavizado"se rellene con más datos la matriz de entrada para que no se pierda la información de los "bordes". Así se consigue que no se pierda la información relativa al primer y último tramo de la señal y no se vean afectados el primero y el último coeficiente.

- **Optimizadores.** Para cada configuración se han ido probando todos los optimizadores que proporciona el módulo Keras, y ante todos, el que mejores resultados ha dado en la mayoría de las ejecuciones ha sido el optimizador Adamax. Para la configuración dada, estos son los resultados con cada optimizador.

Optimizador	Precisión
Adam	0.707142
Adamax	0.718452
Adadelta	0.706547
Nadam	0.640476
Rmsprop	0.691071
SGD	0.511309
Adagrad	0.523809

Tabla 6.4: Resultados de la configuración final CNN con todos los optimizadores. Elaboración propia.

  Como se aprecia, los optimizadores que mejor funcionan son Adam, Adamax y Adadelta, prevaleciendo por muy poco Adamax.

- **Batch size.** El tamaño de batch indica de cuántos en cuántos inputs se entrena la red neuronal. Con batch_size = 32, lo que significa que la red coge 32 inputs y entrena el sistema; después coge los siguientes 32 y vuelve a entrenar, y así hasta completar el entrenamiento con todos los datos.

  Un tamaño de batch más pequeño implica que al principio la precisión crece muy lenta y conforme más va entrenando crece más rápido, mientras que un tamaño más grande (generalmente de 64 o 128 en adelante) implica un crecimiento más rápido en la precisión inicialmente y posteriormente más lento. Tras probar distintos valores, 32 se adapta bastante bien al sistema, puesto que de los valores más bajos que se suelen dar (8, 16 y 32) es el que mejores resultados da, y a partir de 64 empeora considerablemente la precisión del sistema.

- **Epochs.** Las epochs son el número de iteraciones de entrenamiento que realiza la red neuronal, aunque inicialmente se empieza (como se aprecia en la descripción de la sección

6.3) con 100 epochs en la CNN, dependiendo del número total de parámetros que se obtengan de los audios, es necesario modificar este valor.

Experimentalmente se ha comprobado que para el sistema, 72 son las epochs necesarias para tener el modelo entrenado y obtener buenos resultados sin llegar al sobreentrenamiento de la red.

Capítulo 7

Experimento 4 (MFCCs y mejoras)

En este capítulo se partirá del resultado final del experimento 3 y, manteniendo la configuración y estructura del modelo, se aplicarán técnicas de procesamiento de audio y tratamiento de las características, con el fin de mejorar los resultados del experimento anterior.

7.1. Marco teórico

Una vez se tiene una configuración que obtiene unos resultados relevante, el siguiente experimento consistirá en introducir mejoras para reforzar los MFCCs. Debido a su construcción, los MFCCs no son muy robustos frente al ruido, por lo que es necesario realizar algún procesamiento o tratamiento del ruido en los audios.

Durante este experimento se pondrán a prueba todas las técnicas utilizadas por otros artículos científicos que muestran que mejoran los resultados y se comprobará cómo de eficaces suelen ser estas técnicas. Además, ya que el método utilizado en este estudio es muy similar a las técnicas utilizadas en SR, se importarán algunas de ellas para comprobar si funcionan también en el reconocimiento de emociones. Lo interesante es intentar mejorar aún más los resultados obtenidos con la mejor configuración del experimento anterior, fortaleciendo los datos de entrenamiento.

- **Pre-énfasis.** En inglés, pre-emphasis, es un filtro que se le aplica a la señal para

amplificar las frecuencias más altas, es útil por varias razones:

1. Evita algunos problemas numéricos a la hora de realizar la Transformada de Fourier.
2. Balancea el espectro de frecuencias ligeramente, otorgando una mayor magnitud a los agudos, que generalmente quedan sobrepasados por los graves.
3. Mejora el SNR (Signal-t-Noise Ratio), es decir, lo vuelve más robusto al ruido.

La fórmula que define el pre-emphasis para una señal x, para un instante de tiempo t, es la siguiente:

$$y(t) = x(t) - \alpha x(t-1)$$

Generalmente α suele oscilar entre los valores 0,97 y 0,95. El pre-énfasis suele aplicarse a la señal antes de transformarla o realizar ninguna operación sobre ella. Este valor de α indica cuánto se enfatiza la señal.

- **Reducción de ruido con Spectral gating.** Este es un algoritmo de reducción de ruido, basado en el homónimo desarrollado por Audacity (sin reproducirlo completamente). Este algoritmo necesita dos inputs: el audio al que eliminar el ruido y un fragmento de muestra del ruido a eliminar. Generalmente se aplica a audios en los que el ruido de fondo es común y es sencillo de encontrar una muestra de dicho ruido. Aún así, se le puede pasar también un fragmento del propio audio en el que se escuche el ruido únicamente y funciona de igual manera.

  El principio en el que se basa es el siguiente: se aplica la Transformada de Fourier al fragmento de audio que contiene el ruido y se obtienen estadísticas sobre esta para calcular un umbral. Se realiza la FFT a la señal de audio original y se crea una máscara comparando la señal de la FFT al umbral. Se pule la máscara con un filtro, se invierte y se aplica a la señal original de audio.

  Este parte de la ley física de que dos ondas iguales desfasadas π radianes (una es la inversa de la otra) al chocar se anulan. Al determinar el umbral, se determinan cuáles son las

ondas (por frecuencias) que producen el ruido. Cuando está determinado el umbral se aplica a la señal para obtener las frecuencias (y por ende las ondas) que forman el ruido, las cuales formarán la máscara. De igual manera, si se invierte la máscara y se aplica a la señal, al ser ondas inversas, se destruyen haciendo desaparecer el ruido.

Este algoritmo viene implementado en un módulo de Python llamado noisereduce, lo que facilita bastante su aplicación al sistema. Generalmente, la limpieza de ruido se aplica en primer lugar al audio, antes de modificar la señal o aplicar filtros como el pre-énfasis.

- **Liftering sinusoidal.** Técnica introducida en 1987 [32] para el Speech Recognition. Esta técnica no aparece en ningún artículo de reconocimiento de emociones, pero en varias referencias webs se encuentra esta técnica aplicada al caso de estudio, afirmando que mejora los resultados, como por ejemplo la página Practical Cryptography, en un tutorial de MFCCs [33].

  La técnica consiste en la aplicación de un filtro al resultado de la FFT sobre la onda que regula la onda convirtiendo los picos de energía en variaciones y transiciones más suaves. La fórmula del filtro a aplicar, para una ventana w(k) de la señal transformada es:

$$w(k) = 1 + 6sin(\pi k/L)$$

  siendo k cada frecuencia de la ventana w, y L un parámetro que indica cuánto se suaviza la onda. En la siguiente sección se matizará sobre su aplicación y el valor de L.

- **Normalización de MFCCs.** Otra manera de mejorar el SNR consiste en restar a cada coeficiente la media de su valor en todos los tramos en los que se divide la señal.

- **Speaker Normalization.** Es una técnica de normalización de audio que citan varios artículos [34, 35] pero con el inconveniente de que ninguno de ellos muestra cómo se aplica, simplemente indican que lo han aplicado y que mejoran sus resultados. Una fuente de información donde indica cómo se aplica esta técnica a una base de datos de audios, es la tesis doctoral del Dr. Siqing Wu [36]. En esta tesis propone 3 tipos de aplicación

de SN (speaker normalizatio):

1. La primera es aplicar una fórmula para un tipo de características concretas sobre el audio, utilizando la varianza y la media de dichas características.
2. Similar a la anterior propuesta, pero en vez de utilizar la varianza y la media, se utilizan las características del estado neutro como referencia.
3. Las características extraídas son directamente escaladas al intervalo [-1, 1].

Sin embargo, no existen librerías que permitan ejecutar estas técnicas sobre los audios. Esto se debe a que, al igual que en dicha tesis se propone 3 maneras de realizar SN , depende del problema a utilizar y del procesamiento que lleven los datos y características.

En nuestro caso no se realiza un proyecto o un desarrollo tan complejo como el que se muestra en la tesis, por lo que, al no extraer tantas características además, no merece la pena integrarlo en el proyecto. Para integrarlo y probarlo sería necesario rehacer toda la toma de características, añadiendo nuevas y cambiando el planteamiento que se tenía desarrollado hasta ahora, saliéndose de los ámbitos y límites del proyecto. Aun así, es una de las tareas que podría realizarse en trabajos futuros.

7.2. Modelo, ejecución y análisis de resultados

Una vez definidas las técnicas, se procede a su integración en el proyecto por orden de aplicación en el sistema: procesamiento de audio, extracción de características y construcción del modelo.

7.2.1. Reducción de ruido

En primer lugar se comienza por el algoritmo de reducción de ruido. Debido a que cada audio conformante de la base de datos está grabado por una persona distinta, es difícil determinar un ruido común para todos. Además, durante la estructuración y organización de

la base de datos, se comprobó que todos los audios tenían un ruido ambiente prácticamente nulo, estaban grabados con bastante buena nitidez.

Esto no implica que ese argumento sea válido para justificar que no necesitan el algoritmo. Debido a que al principio del audio se dejaban unos 100 ms antes de que comenzara la frase, se aplica el algoritmo a todos los audios, indicando como fragmento de ruido los 100 primeros ms de cada archivo.

Sin reducción de ruido	Con reducción de ruido
0.718452	0.716071

Tabla 7.1: Resultados con y sin reducción de ruido.

El resultado es prácticamente idéntico que al no aplicarlo, por lo que se puede confirmar que los audios de la base de datos están limpios de ruido.

7.2.2. Pre-énfasis

La siguiente técnica que se va a tratar es el pre-énfasis. esta técnica será aplicada directamente a los audios antes de calcular los MFCCs, se aplica con los valores para α de 0,97 y 0,95

$\alpha = 0{,}97$	$\alpha = 0{,}95$
0.754761	0.723214

Tabla 7.2: Resultados con distintos valores para pre-énfasis.

Como se aprecia, realmente el pre-énfasis es una medida que mejora los resultados, con una notable influencia en la precisión total. Partiendo de la teoría que lo rodea, realmente enfatizar las frecuencias más agudas es un acierto, puesto que hay información relevante que se estaba obviando.

7.2.3. Liftering sinusoidal

La siguiente técnica que se aplicaría sería el liftering sinusoidal. Como se ha mencionado, la aplicación de este suavizado es convertir los picos de energía en valores más acordes a la tendencia de los datos. Se aplica a los pequeños tramos de la señal (cuando se divide) mediante la fórmula anteriormente indicada.

Sin embargo hay un parámetro L que el valor que tiene fluctúa dependiendo de dónde se consulte. El artículo padre de este concepto [32] declara en él que, experimentalmente, utilizan el valor L=12 ya que es el que mejor resultados les ofrece. Ante tal afirmación, se realizaron varias pruebas con distintos valores para L:

Valor de L	Precisión del sistema
12	0.705357
17	0.688095
22	0.710714
num_MFCCs	0.658333
num_MFCCs * 2	0.621428
num_MFCCs / 2	0.718452

Tabla 7.3: Resultados con distintos valores para L usando liftering sinusoidal.

Algunos de los valores están expresados en función del número de MFCCs que se escojan, ya que el artículo original hace referencia a que obtienen el valor de L a partir de los coeficientes que calculan, y una de las pruebas fue utilizar esos valores concretamente: el doble, la mitad y el mismo número. En nuestro caso, los resultados no sólo no mejoran, sino que nos restan la mejora conseguida con el pre-énfasis.

La respuesta a esto se debe que el liftering elimina los cambios bruscos de energía, y por tanto suavizando mucho la entonación. Esto proporciona una limpieza clara de la información en lo que se refiere a SR, pero en el ámbito de las emociones, elimina una parte de la información que puede ser crucial a la hora de reconocer alguna emoción.

Por definición, esta técnica perjudica, sobre todo, el reconocimiento de aquellas emociones

que se caractericen por tener altos picos de energía, como el miedo (en algunos casos) y la sorpresa.

7.2.4. Normalización de los MFCCs.

La normalización de los MFCCs proporciona muy buenos resultados en SR, pero en el reconocimiento de emociones. Aplicándolo tal y como se describe en la sección 7.1, disminuye la precisión del sistema hasta 0.717857 cuando también se ha aplicado pre-énfasis.

De la misma forma que el caso anterior, resta lo ganado por la enfatización de la onda, ya que produce un resultado muy similar. Al normalizar los coeficientes, se igualan más los valores, perdiéndose la información relativa a cambios bruscos de energía en la entonación, perjudicando a las mismas emociones que el caso anterior. Por tanto, aunque es bueno para SR, no lo es para el reconocimiento de emociones.

7.2.5. Resultados

Una vez realizados los experimentos aplicando todas estas técnicas, queda claro que la que mejora la precisión total del sistema es el uso del pre-énfasis con α igual a 0,97. Por tanto, se realiza una ejecución completa, aplicando también el 10-fold cross validation y creando unas nuevas particiones para dicha ejecución. Además, se ofrecerán unas medidas más exhaustivas conforme a la calidad y precisión del sistema. Estas medidas serán las popularmente conocidas en el ámbito del PNL llamadas recall, precisión y F1-score. Para ello es necesario calcular tres valores: falsos positivos (FP, audios no pertenecientes a una emoción pero clasificados como tal emoción); falsos negativos (FN, audios pertenecientes a una emoción pero no clasificados como tal emoción); y verdaderos positivos (TP, audios pertenecientes a una emoción y clasificados como tal emoción). Asimismo, se anotarán, para cada una de las emociones, cuántos audios son clasificados correctamente y cuántos se clasifican como otra emoción.

- **Precision.** Dentro del ámbito de la clasificación, que es el que ocupa a este proyecto, la precisión mide, para una categoría, la cantidad de elementos correctamente clasificados

dentro de esa categoría.

- **Recall.** Muestra el ratio real de los elementos clasificados pertenecientes a una misma categoría. Relacionado con la sensibilidad, mide el número de elementos clasificados correctamente con respecto al número total de elementos pertenecientes a esa clase.

- **F1-score.** En castellano Valor-F, mide la precisión total del sistema combinando las dos métricas anteriores. Su valor se calcula realizando la media armónica entre Precision y Recall.

Los resultados se mostrarán a continuación en la siguiente matriz de confusión, tabla 7.4 , conteniendo la clasificación de los audios en cada una de las categorías, las métricas para cada una de las emociones y las métricas para el total. La lectura de las emociones de la tabla se lee por filas, de manera que cada pareja fila-columna se interprete como cuántos audios de la categoría real fila con clasificados como la emoción columna.

Emociones	Alegría	Enfado	Tristeza	Miedo	Sorpresa	Asco	Neutro
Alegría	173	$10^{(1)}$	6	12	$28^{(2)}$	5	6
Enfado	$19^{(1)}$	183	11	6	$13^{(3)}$	6	2
Tristeza	4	10	200	8	1	11	6
Miedo	11	8	11	163	14	$27^{(4)}$	6
Sorpresa	$30^{(2)}$	$18^{(3)}$	2	9	173	5	3
Asco	11	11	8	$28^{(4)}$	8	161	13
Neutro	10	7	9	7	2	9	196

Tabla 7.4: Clasificación de los audios en las distintas emociones. Elaboración propia.

En la tabla 7.4 se remarca, en un color azul claro, el número de audios que son clasificados correctamente en la emoción a la que pertenecen, distribuidos en la diagonal principal de la tabla. Observándola, destacan las emociones de tristeza y neutro como las mejor clasificadas (menor error), seguidas de alegría, enfado y sorpresa, y consiguiendo los peores resultados, miedo y asco.

De esta primera observación se puede extraer que las emociones con una expresividad más marcada y característica son tristeza y neutro, por lo que son las más reconocibles; al

contrario, en cuanto a las emociones de miedo y asco, al tener un ratio de acierto más bajo nos indican que la expresividad es más variable, existiendo diversas formas de mostrarlas y siendo más difícilmente reconocibles.

También se han recalcado, mediante super índices del 1 al 4 , otros datos mal clasificados. Estos datos están seleccionados por pares, simétricos con respecto a la diagonal principal, de manera que representan aquellos audios clasificados erróneamente de categoría, pero que ocurre de manera similar a la inversa. En otras palabras, relacionan las dos emociones más frecuentemente confundidas por el sistema; así pues, se observa que 28 audios de alegría son clasificados como sorpresa y 30 de sorpresa son catalogados como alegría. Dichos pares de emociones más confundidos entre sí son asco-miedo y alegría-sorpresa, y en menor medida, alegría-enfado y enfado-sorpresa. Esto implica que dichas emociones en muchos audios son expresadas, dentro de las variaciones que permiten, con una entonación casi idéntica. En cuanto a la alegría y sorpresa, se debe a que la sorpresa puede ser tanto positiva como negativa, según la atracción o aversión que le produzca el evento, objeto o situación que provoque el estímulo [37]. Así pues, dado que la alegría suele expresarse con mayor énfasis y volumen (energía), esta puede ser confundida fácilmente con una sorpresa positiva, justificándose el similar número de audios mal clasificados entre ambas emociones.

Por las mismas razones, la sorpresa negativa puede confundirse con el enfado, pero siendo este caso menos frecuente que el anterior por ligeros matices en la entonación (y no tanto en la energía con la que se exprese la frase) que resultan lo suficientemente característicos para no mezclarlos. Al producirse ambas emociones (principalmente por estímulos que producen una reacción negativa en la persona) pueden ser en algunos casos hasta idénticas. No obstante, este trabajo también ratifica la diferenciación entre ambas, debido a la entonación de cada voz.

La tabla 7.5 no hace sino confirmar mediante métricas más completas y específicas, los resultados apreciables en la tabla 7.4 . Dado que las métricas utilizadas se aplican en sistemas binarios, para su aplicación a los datos que se tienen, se ha considerado la transformación de

Emociones	FP	FN	TP	Precision	Recall	F1-score
Alegría	85	67	173	0,6705426357	0,7208333333	0,6947791165
Enfado	64	57	183	0,7408906883	0,7625	0,7515400411
Tristeza	47	40	200	0,8097165992	0,8333333333	0,8213552361
Miedo	70	77	163	0,6995708155	0,6791666667	0,689217759
Sorpresa	66	67	173	0,7238493724	0,7208333333	0,7223382046
Asco	63	79	161	0,71875	0,6708333333	0,6939655172
Neutro	36	44	196	0,8448275862	0,8166666667	0,8305084746
Total	431	431	1249	0,743452381	0,743452381	0,743452381
Media	-	-	-	0,744021099	0,743452381	0,743386335

Tabla 7.5: Métricas de la ejecución final: Precision, Recall y F1.

categorías de la siguiente manera: un audio puede pertenecer a la emoción n o no.

Este planteamiento permite convertir un sistema de clasificación multiclase en n sistemas de clasificación binarios. Para cada uno se tomará el número de audios pertenecientes a dicha emoción (240) como la parte de la hipótesis un audio pertenece a la emoción n, y el número restante de audios (1440) como la parte un audio no pertenece a la emoción n.

Para calcular estas métricas en un sistema de clasificación multiclase como el que se desarrolla en este proyecto, es necesario tener en cuenta si las clases están balanceadas en cuanto a número de elementos o no. Si no lo están, existen variantes de estas métricas: con ponderación (micro) y sin ponderación (macro) . Se calculan realizando la media de las métricas de cada una de las clases por separado (como se ha explicado en el párrafo anterior), utilizando como pesos el número de elementos de cada clase. La media ponderada se utiliza para equilibrar los resultados cuando las clases no están balanceadas, obteniendo un resultado representativo de todas las categorías . Como consecuencia, cuando las clases tienen el mismo número de objetos, entonces ambas clases coinciden, por lo que únicamente se muestra una vez el valor en la tabla 7.5.

Otra manera de calcular las métricas generales para el sistema, es realizar una suma de todos los valores (FN, FP y TP), y realizar los cálculos conforme a estos datos. En

la tabla 7.5 se reflejan ambos métodos, siendo los resultados relativos al último explicado reflejados en la fila con etiqueta Total, y los resultantes al uso de la media mostrados con Media

En ambos casos nos muestra unos resultados bastante similares, lo que refleja el balance de elementos distribuidos en las categorías. Además, tanto en cada categoría como en general del sistema, los valores de Precision y Recall son muy cercanos. Esto indica que el sistema tiene una buena tendencia a clasificar los audios correctamente y a acertar en un gran número de ellos; en conclusión, se puede afirmar que el sistema es fiable y preciso.

7.3. Conclusiones y reajustes

De las técnicas descritas y aplicadas al sistema general, se han comprobado la validez y eficacia para aplicarse al reconocimiento de emociones o se ha analizado bien para demostrar por qué no es una buena herramienta para dicho campo de estudio. Aunque todavía existen muchas más técnicas para aplicar, al igual que distintos tipos de características que extraer de los audios, se ha decidido no incluirlas en el presente proyecto.

La opción elegida para desarrollar finalmente han sido los MFCCs ya que ofrecen características bastante buenas para este tipo de problemas, a la vez que, como ha quedado demostrado en esta sección, contienen una numerosa variedad de técnicas para pulirlos y resaltar un tipo de valores u otros. El hecho de que las técnicas que mejoran los resultados en SR, como el liftering y la normalización de los MFCCs, no lo hagan para la detección de emociones, es debido a que dichas técnicas enfatizan y atenúan unas características más aptas para SR que para la detección de emociones. Esto es porque mediante esas técnicas se atenúan las características más relevantes y se enfatizan las que no lo son tanto, perdiendo la información en el proceso.

No obstante, se concluye que la solución ofrecida por el sistema construido es buena y precisa (como se ha demostrado con los resultados de las métricas) consiguiendo, de media,

una precisión de acierto cercana 75 % mediante el uso de una red neuronal CNN.

Capítulo 8

Conclusiones y trabajos futuros

Finalmente, tras haber realizado todos los experimentos y analizado los resultados, consiguiendo una buena solución, en el presente y último capítulo, se realizan unas conclusiones generales del proyecto, los trabajos futuros a los que da pie y una valoración personal.

8.1. Resumen y conclusiones

El mayor reto a la hora de afrontar este estudio, era construir un sistema basado en un modelo de redes neuronales, que obtuviera buenos resultados. Existen numerosos artículos que afrontan el problema de la detección de emociones a partir del tono de voz, mediante el uso de clasificadores, consiguiendo unos resultados bastante altos. No obstante, existe una gran brecha en cuanto a la precisión de acierto obtenida por los clasificadores frente a la obtenida por las redes neuronales, proporcionando, estas últimas, resultados más bajos. Esto se puede ver en los resultados proporcionados por Kerkeni et al. [34], en los que se muestra que, con las mismas características, los el clasificador usado (SVM) da mejores resultados que la red neuronal. No obstante, como el objetivo inicial del proyecto era su aplicación en redes neuronales, no se ha tomado como referencia los resultados que pudiera ofrecer un clasificador.

Inicialmente, con los dos primeros experimentos, se tomaron ideas propias para llevarlas a cabo, de modo que, si resultaban exitosas, se ofreciesen nuevas vías de exploración y enfoque

de este tema. Sin embargo, aunque finalmente no funcionaron, sirvieron para estudiar por qué fallaban y así dar con las características adecuadas más rápidamente, sabiendo por qué sí iban a funcionar: los coeficientes de Mel.

Gracias a los MFCCs, se ha podido construir un sistema robusto, con una buena precisión que cumple con el objetivo propuesto para este proyecto. Del mismo modo, se ha proporcionado también un análisis detallado de los resultados obtenidos, permitiendo así poder realizar más matices y reajustes en futuras investigaciones y versiones del proyecto.

No obstante, aunque las técnicas de refinamiento de los MFCCs probadas han sido variadas, únicamente se ha escogido por aplicar una de ellas: el pre-énfasis. El resto de técnicas se desecharon por los motivos explicados en el capitulo 7, por lo que, en cierto modo, se ha construido un modelo con características prácticamente en bruto. Existen aún más técnicas que no se han aplicado al sistema, por sobrepasar los límites de aplicación y estudio del proyecto, pero que permitirían refinar las características, y mejorar notablemente los resultados. Aun así, a pesar del poco procesamiento final aplicado, se consiguen unos resultados de un 75 %, concluyendo en una buena solución para el problema planteado.

El proyecto, como trabajo de investigación, no se ha centrado únicamente en la descripción de una solución que funcione, sino en detallar todo el proceso seguido hasta la consecución de la misma; de modo que, leyendo dicho estudio, permita entender por qué unas características son mejores que otras y por qué funcionan. Otro de los objetivos que se pretendía conseguir con este proyecto, es la reproducibilidad del mismo, ofreciendo todos los razonamientos y herramientas para que se pueda construir el mismo sistema, aún con datos propios, y corroborar los resultados aquí ofrecidos.

8.2. Trabajos futuros

Como acabo de expresar, la investigación no tiene un punto y final, únicamente capítulos que se acaban. Y que un sistema funcione, no quiere decir que no se pueda mejorar. Las vías que ofrece este proyecto son muchas:

- Desarrollo de un sistema híbrido de detección de emociones en la entonación y mediante PLN. Crear un sistema híbrido, permitiría explorar más matices de la expresión de las emociones, vocalmente hablando. Esto permitiría entrar a dar una solución al tratamiento de la ironía o el sarcasmo en PLN.
- El sistema aún puede mejorarse estudiando y aplicando las técnicas de Speaker Normalization. También puede hacerse una revisión para hacer un tratamiento más específico de los MFCCs, etc.
- Se puede ampliar a varios idiomas, para que no sea únicamente específico del castellano, de esta manera tendríamos un sistema universal, capaz de reconocer la emoción independientemente del idioma.
- Hacer un estudio de la base de datos para ampliar las edades de las personas o conseguir incluir más emociones.

Debido a que esta rama de investigación es relativamente nueva, y permite su integración e hibridación con otras como PLN, SR, etc. así como realizar mejoras más ambiciosas de este proyecto, los trabajos que permite este área pueden mostrar aún muchos descubrimientos y realizar avances más complejos.

8.3. Valoración personal

El proyecto desarrollado como Trabajo Final de Máster ha sido todo un reto personal en el que se ha tenido que poner en práctica los conocimientos, aptitudes y habilidades para poder llevarlo adelante. Además la temática ha sido algo totalmente nuevo a lo que no me había enfrentado antes, que es un proyecto de investigación. Ha sido un camino completo desde

una idea que surgió de realizar este proyecto hasta completarlo, sin siquiera saber muchas veces cuál es el camino.

Me ha servido para darme cuenta lo sacrificada que es la investigación: por cada vez que encuentras la manera correcta de conseguir algo, descubres mil maneras de hacerlo mal. Y a veces puede ser frustrante, porque en muchas ocasiones, la respuesta correcta tarda en aparecer.

El principal problema que he tenido durante el desarrollo de este proyecto ha sido ponerme límites. Vivimos en un mundo con tal cantidad de conocimiento, que cuando crees saberlo todo sobre un tema, siempre aparecerán nuevos detalles que no conoces. De igual manera, cuando crees que no se puede mejorar más algo que has hecho, siempre aparecen nuevos detalles que se pueden aplicar, nuevas teorías o artículos que no habías visto 6 meses antes, etc.

En mi caso, me puse el objetivo de intentar conseguir que superara el 70 % de precisión con este proyecto, y hasta que no lo conseguí, no paré. Y aún así, cuando llegué, seguí probando cosas por ver si seguía mejorando. Me ha servido para saber ponerme metas y objetivos alcanzables y no ser demasiado ambicioso, de manera que lo que se ofrezca, aunque no sea lo más novedoso de la historia y no contenga todas las técnicas existentes de la temática, sea de calidad y, sobre todo, sabiendo y entendiendo lo que se ha hecho y conseguido.

Capítulo 9

Bibliografía

[1] Joseph CR Licklider. Man-computer symbiosis. IRE transactions on human factors in electronics, (1):4–11, 1960.

[2] Joseph Carl Robnett Licklider and Welden E Clark. On-line man-computer communication. In Proceedings of the May 1-3, 1962, spring joint computer conference, pages 113–128, 1962.

[3] Gobinda G Chowdhury. Natural language processing. Annual review of information science and technology, 37(1):51–89, 2003.

[4] Ali Yadollahi, Ameneh Gholipour Shahraki, and Osmar R Zaiane. Current state of text sentiment analysis from opinion to emotion mining. ACM Computing Surveys (CSUR), 50(2):1–33, 2017.

[5] Paul Ekman, Wallace V Friesen, and Phoebe Ellsworth. Emotion in the Human Face: Guide-lines for Research and an Integration of Findings: Guidelines for Research and an Integration of Findings. Pergamon, 1972.

[6] Paul Ekman. An argument for basic emotions. Cognition & emotion, 6(3-4):169–200, 1992.

[7] Robert Plutchik and Henry Kellerman. Theories of emotion, volume 1. Academic Press, 2013.

[8] Phillip Shaver, Judith Schwartz, Donald Kirson, and Cary O'connor. Emotion knowledge: further exploration of a prototype approach. Journal of personality and social psychology, 52(6):1061, 1987.

[9] Hugo Lövheim. A new three-dimensional model for emotions and monoamine neurotransmitters. Medical hypotheses, 78(2):341–348, 2012.

[10] Trevor R Agus, Simon J Thorpe, Clara Suied, and Daniel Pressnitzer. Characteristics of human voice processing. In Proceedings of 2010 IEEE International Symposium on Circuits and Systems, pages 509–512. IEEE, 2010.

[11] Felix Burkhardt and Walter F Sendlmeier. Verification of acoustical correlates of emotional speech using formant-synthesis. In ISCA Tutorial and Research Workshop (ITRW) on speech and emotion, 2000.

[12] William Roberts and Janet Strayer. Empathy, emotional expressiveness, and prosocial behavior. Child development, 67(2):449–470, 1996.

[13] Wahab Khan, Ali Daud, Jamal A Nasir, and Tehmina Amjad. A survey on the state-of-the-art machine learning models in the context of nlp. Kuwait journal of Science, 43(4), 2016.

[14] Kun Han, Dong Yu, and Ivan Tashev. Speech emotion recognition using deep neural network and extreme learning machine. In Fifteenth annual conference of the international speech communication association, 2014.

[15] Valery A Petrushin. Detecting emotions using voice signal analysis, May 22 2007. US Patent 7,222,075.

[16] Iker Luengo, Eva Navas, Inmaculada Hernáez, and Jon Sánchez. Reconocimiento automático de emociones utilizando parámetros prosódicos. Procesamiento del lenguaje natural, (35):13–20, 2005.

[17] Ildar Batyrshin and G Sidorov. Advances in artificial intelligence. Lecture notes in computer science, 2012.

[18] Santiago Planet Garcia, Jose Antonio Morán Moreno, and Lluís Formiga Fanals. Reconocimiento de emociones basado en el análisis de la señal de voz parametrizada.

[19] Felix Burkhardt, Astrid Paeschke, Miriam Rolfes, Walter F Sendlmeier, and Benjamin Weiss. A database of german emotional speech. In Ninth European Conference on Speech Communication and Technology, 2005.

[20] Juan M López, Idoia Cearreta, Nestor Garay, K López de Ipiña, and Andoni Beristain. Creación de una base de datos emocional bilingüe y multimodal. In Redondo, MA, Bravo C., Ortega M.(Eds). Proceeding of the 7th Spanish Human Computer Interaction Conference, Interacción, volume 6, pages 55–66, 2006.

[21] Houwei Cao, David G Cooper, Michael K Keutmann, Ruben C Gur, Ani Nenkova, and Ragini Verma. Crema-d: Crowd-sourced emotional multimodal actors dataset. IEEE transactions on a□ective computing, 5(4):377–390, 2014.

[22] Steven R Livingstone and Frank A Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. PloS one, 13(5):e0196391, 2018.

[23] Carlos Busso, Srinivas Parthasarathy, Alec Burmania, Mohammed AbdelWahab, Najmeh Sadoughi, and Emily Mower Provost. Msp-improv: An acted corpus of dyadic interactions to study emotion perception. IEEE Transactions on A□ective Computing, 8(1):67–80, 2016.

[24] Juanjo Igartua, Javier Álvarez, José Antonio Adrián, and Darío Páez. Música, imagen y emoción: una perspectiva vigotskiana. Psicothema, 6(3):347–356, 1994.

[25] Avery Wang et al. An industrial strength audio search algorithm. In Ismir, volume 2003, pages 7–13. Washington, DC, 2003.

[26] Pacific Northwest Seismic Network. What is a spectrogram? urlhttps://pnsn.org/spectrograms/what-is-a-spectrogram. Accedido 3-07-2020.

[27] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386, 1958.

[28] Marvin Minsky and Seymour A Papert. Perceptrons: An introduction to computational geometry. MIT press, 2017.

[29] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735–1780, 1997.

[30] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998.

[31] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[32] Biing-Hwang Juang, L Rabiner, and JG Wilpon. On the use of bandpass liftering in speech recognition. IEEE Transactions on acoustics, speech, and signal processing, 35(7):947–954, 1987.

[33] Practical Cryptography. Mel frequency cepstral coefficient (mfcc) tutorial. http://practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coeffic

[34] Leila Kerkeni, Youssef Serrestou, Mohamed Mbarki, Kosai Raoof, Mohamed Ali Mahjoub, and Catherine Cleder. Automatic speech emotion recognition using machine learning. In Social Media and Machine Learning. IntechOpen, 2019.

[35] Vidhyasaharan Sethu, Eliathamby Ambikairajah, and Julien Epps. Speaker normalisation for speech-based emotion detection. In 2007 15th international conference on digital signal processing, pages 611–614. IEEE, 2007.

[36] Siqing Wu. Recognition of human emotion in speech using modulation spectral features and support vector machines. PhD thesis, 2009.

[37] Paul Ekman and Wallace V Friesen. Unmasking the face: A guide to recognizing emotions from facial clues. Ishk, 2003.

Capítulo 10

Anexo I

ALEGRÍA

1. Hoy va a ser un gran día, lo presiento.
2. ¡Por fin lo he conseguido!
3. Ya estamos todos juntos por fin.
4. No me puedo creer que me haya dicho que sí.
5. Me lo estoy pasando estupendamente con vosotros esta noche.
6. He conseguido que <inserta cantante que te guste>me firme su disco.
7. Me acaban de dar la mejor noticia de mi vida, ¡no puedo ser más feliz!
8. Después de muchas horas, por fin he terminado el informe.
9. Lo importante es perseverar.
10. Qué alegría volver a verte después de tanto tiempo.
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

ENFADO

1. ¿Por qué creemos que es necesario decir gilipolleces para estar cómodos?
2. Lo que pasa aquí es que falta comunicación.
3. No me apetece hablar con un imbécil como tú.
4. Me acabo de pillar los dedos con la puerta.
5. No os imagináis lo que puedo llegar a odiar los días de lluvia.
6. Haciendo trampas, no vuelvo a jugar contigo.
7. No te pienso prestar más dinero hasta que no me devuelvas el que me debes.
8. Ni se te ocurra volver a hacer eso.
9. Qué pesado/a eres, déjame en paz.
10. Últimamente no tengo tiempo para nada en el trabajo y me agobio.
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

TRISTEZA

1. Quiero estar sola/o.
2. Hoy te he vuelto a recordar.
3. Dice que me quiere pero solo como amigo/a.
4. Al llegar a casa es cuando me doy cuenta de que he perdido las llaves.
5. No tengo ganas de nada.
6. No te pido que me perdones, solamente que me entiendas.
7. Al final no he conseguido pasar las pruebas.
8. Te voy a echar muchísimo de menos.
9. Tengo que abandonar este proyecto hasta que pueda estar al 100
10. Así no puedo.
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

SORPRESA

1. ¡Vaya giro que ha dado la serie en el último episodio!
2. Me acabo de encontrar a <insertar famoso/a>en la puerta de mi casa.
3. Ya se me ha vuelto a olvidar.
4. Increíble esto, vaya pasada.
5. ¡No me lo puedo creer!
6. Mandé un correo hace cinco minutos y ya me han contestado.
7. Lo daba por perdido y mira lo que me he encontrado.
8. No esperaba verte aquí.
9. No me creo que hayas vuelto a acertar.
10. ¡Otra vez lo mismo!
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

ASCO

1. Somos todos maravillosos, y todos damos asco.
2. Hemos salido fuera y me han puesto <insertar comida que te dé asco>para comer.
3. No es que te odie, es que me das asco.
4. Eres asqueroso.
5. Podrías cambiarte de ropa más a menudo.
6. No puedo entrar del mal olor que hay.
7. No he visto nada más desagradable en mi vida.
8. Me dan náuseas solo de pensarlo.
9. (Tío/a,)por favor, no vuelvas a hacer eso.
10. Soy incapaz de cambiarle el pañal al bebé.
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

MIEDO

1. La cosa debajo de mi cama que está esperando para agarrarme el tobillo no es real.
2. ¡Eso no, cualquier cosa menos eso!
3. No encuentro la cartera con todo el dinero dentro.
4. Por favor, que no me escuche.
5. Tengo 13 llamadas perdidas de mi madre.
6. Creo que no he guardado antes de cerrar el word.
7. ¿Qué ha sido ese ruido?
8. No tengo ni idea de cómo va a salir esto.
9. ¡No te acerques a mi!
10. Me da un miedo que mi dedo no se recupere...
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.

13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.

NEUTRO

1. Cuando comience a hervir, añádele la gelatina de limón y el azúcar.
2. En la naturaleza tenemos cuerpos con carga positiva, negativa y neutra.
3. Se llama grado a cada una de las notas de la escala.
4. La teoría del aprendizaje significativo supone poner de relieve el proceso de construcción de significados como elemento central de la enseñanza.
5. La faringitis es la inflamación de la garganta o faringe a menudo causada por una infección bacteriana o vírica.
6. Spotify es una aplicación multiplataforma sueca, empleada para la reproducción de música vía streaming.
7. El arte medieval es una etapa de la historia del arte que cubre un prolongado período para una enorme extensión espacial.
8. Viaje al centro de la Tierra es una novela de Julio Verne, publicada el 25 de noviembre de 1864, que trata de la expedición de un profesor de mineralogía, su sobrino y un guía al interior de la Tierra.

9. Todo apunta a un repunte de precios del aceite de oliva.
10. Una persona tranquila no siente preocupación.
11. No te olvides de dónde vienes.
12. Hoy para comer tenemos albóndigas en salsa de tomate con guisantes.
13. Sólo quiero ir a cualquier otra parte.
14. Me han cambiado la cerradura por otra nueva.
15. Está allí, en la habitación 11.
16. Me acaban de pagar el último sueldo.
17. ¿De quién es este bolígrafo que he encontrado?
18. Me ha vuelto a pasar.
19. La película no era lo que me esperaba.
20. Ha pasado la tarde entera sin darme cuenta.